

Photon Fusion

2.0.4

1 Photon Fusion API Documentation	1
1.1 Main Fusion API	1
2 Photon Fusion Overview	3
2.1 Choosing the Right Mode	4
2.2 Topology Differences	5
2.2.1 Server Mode	5
2.2.1.1 Client Side Prediction	5
2.2.2 Host Mode	6
2.2.3 Shared Mode	6
2.2.4 Cost	6
3 Hierarchical Index	7
3.1 Class Hierarchy	7
4 Class Index	15
4.1 Class List	15
5 Namespace Documentation	33
5.1 Fusion Namespace Reference	33
5.1.1 Enumeration Type Documentation	47
5.1.1.1 CompareOperator	47
5.1.1.2 ConnectionType	47
5.1.1.3 DrawIfMode	47
5.1.1.4 EditorButtonVisibility	48
5.1.1.5 GameMode	48
5.1.1.6 HitboxTypes	48
5.1.1.7 HitOptions	49
5.1.1.8 NetworkObjectAcquireResult	49
5.1.1.9 NetworkObjectFlags	49
5.1.1.10 NetworkObjectHeaderFlags	50
5.1.1.11 NetworkPrefabTableGetPrefabResult	50
5.1.1.12 NetworkSceneInfoChangeSource	51
5.1.1.13 NetworkSceneInfoDefaultFlags	51
5.1.1.14 NetworkSpawnFlags	51
5.1.1.15 NetworkSpawnStatus	52
5.1.1.16 NetworkTypeIdKind	52
5.1.1.17 PageSizes	52
5.1.1.18 PriorityLevel	52
5.1.1.19 RenderSource	53
5.1.1.20 RenderTimeframe	53
5.1.1.21 RpcChannel	53
5.1.1.22 RpcHostMode	54
5.1.1.23 RpcLocalInvokeResult	54

5.1.1.24 RpcSendCullResult	54
5.1.1.25 RpcSendMessageResult	56
5.1.1.26 RpcSources	56
5.1.1.27 RpcTargets	57
5.1.1.28 RpcTargetStatus	57
5.1.1.29 ScriptHeaderBackColor	57
5.1.1.30 ScriptHeaderIcon	58
5.1.1.31 ScriptHeaderStyle	58
5.1.1.32 SessionLobby	58
5.1.1.33 ShutdownReason	58
5.1.1.34 SimulationModes	59
5.1.1.35 SimulationStages	59
5.1.1.36 Topologies	60
5.1.1.37 Units	60
5.1.1.38 UnityPlayerLoopSystemAddMode	61
5.1.2 Function Documentation	61
5.1.2.1 FusionGlobalScriptableObjectUnloadDelegate()	61
5.1.2.2 NetworkObjectSpawnDelegate()	61
5.1.2.3 RpcInvokeDelegate()	61
5.1.2.4 RpcStaticInvokeDelegate()	62
5.2 Fusion.Analyzer Namespace Reference	62
5.2.1 Enumeration Type Documentation	62
5.2.1.1 StaticFieldResetMode	62
5.3 Fusion.Async Namespace Reference	63
5.4 Fusion.Encryption Namespace Reference	63
5.5 Fusion.Internal Namespace Reference	63
5.6 Fusion.LagCompensation Namespace Reference	64
5.6.1 Enumeration Type Documentation	65
5.6.1.1 HitType	65
5.6.2 Function Documentation	65
5.6.2.1 PreProcessingDelegate()	65
5.7 Fusion.Protocol Namespace Reference	66
5.7.1 Enumeration Type Documentation	66
5.7.1.1 DisconnectReason	66
5.8 Fusion.Runtime Namespace Reference	67
5.9 Fusion.Runtime.Unity Namespace Reference	67
5.10 Fusion.Sockets Namespace Reference	67
5.10.1 Enumeration Type Documentation	68
5.10.1.1 NetCommands	68
5.10.1.2 NetConnectFailedReason	68
5.10.1.3 NetConnectionStatus	69
5.10.1.4 NetDisconnectReason	69

5.10.1.5 NetPacketType	69
5.10.1.6 OnConnectionRequestReply	70
5.11 Fusion.Sockets.Stun Namespace Reference	70
5.11.1 Enumeration Type Documentation	70
5.11.1.1 NATType	70
5.12 Fusion.Statistics Namespace Reference	71
5.13 UnityEngine Namespace Reference	71
5.14 UnityEngine.SceneManagement Namespace Reference	71
5.15 UnityEngine.Scripting Namespace Reference	71
5.16 UnityEngine.Serialization Namespace Reference	71
6 Class Documentation	73
6.1 _128 Struct Reference	73
6.1.1 Detailed Description	73
6.1.2 Member Data Documentation	73
6.1.2.1 Data	73
6.1.2.2 SIZE	74
6.2 _16 Struct Reference	74
6.2.1 Detailed Description	74
6.2.2 Member Data Documentation	74
6.2.2.1 Data	74
6.2.2.2 SIZE	74
6.3 _2 Struct Reference	75
6.3.1 Detailed Description	75
6.3.2 Member Data Documentation	75
6.3.2.1 Data	75
6.3.2.2 SIZE	75
6.4 _256 Struct Reference	75
6.4.1 Detailed Description	76
6.4.2 Member Data Documentation	76
6.4.2.1 Data	76
6.4.2.2 SIZE	76
6.5 _32 Struct Reference	76
6.5.1 Detailed Description	77
6.5.2 Member Data Documentation	77
6.5.2.1 Data	77
6.5.2.2 SIZE	77
6.6 _4 Struct Reference	77
6.6.1 Detailed Description	78
6.6.2 Member Data Documentation	78
6.6.2.1 Data	78
6.6.2.2 SIZE	78

6.7 _512 Struct Reference	78
6.7.1 Detailed Description	78
6.7.2 Member Data Documentation	79
6.7.2.1 Data	79
6.7.2.2 SIZE	79
6.8 _64 Struct Reference	79
6.8.1 Detailed Description	79
6.8.2 Member Data Documentation	79
6.8.2.1 Data	80
6.8.2.2 SIZE	80
6.9 _8 Struct Reference	80
6.9.1 Detailed Description	80
6.9.2 Member Data Documentation	80
6.9.2.1 Data	80
6.9.2.2 SIZE	81
6.10 Allocator Struct Reference	81
6.10.1 Detailed Description	82
6.10.2 Member Data Documentation	82
6.10.2.1 BUCKET_COUNT	82
6.10.2.2 BUCKET_INVALID	82
6.10.2.3 HEAP_ALIGNMENT	82
6.10.2.4 REPLICATE_WORD_ALIGN	83
6.10.2.5 REPLICATE_WORD_SHIFT	83
6.10.2.6 REPLICATE_WORD_SIZE	83
6.11 Allocator.Config Struct Reference	83
6.11.1 Detailed Description	84
6.11.2 Constructor & Destructor Documentation	84
6.11.2.1 Config()	84
6.11.3 Member Function Documentation	84
6.11.3.1 Equals() [1/2]	85
6.11.3.2 Equals() [2/2]	85
6.11.3.3 GetHashCode()	85
6.11.3.4 ToString()	86
6.11.4 Member Data Documentation	86
6.11.4.1 BlockCount	86
6.11.4.2 BlockShift	86
6.11.4.3 DEFAULT_BLOCK_COUNT	86
6.11.4.4 DEFAULT_BLOCK_SHIFT	86
6.11.4.5 GlobalsSize	86
6.11.4.6 SIZE	87
6.11.5 Property Documentation	87
6.11.5.1 BlockByteSize	87

6.11.5.2 BlockWordCount	87
6.11.5.3 HeapSizeAllocated	87
6.11.5.4 HeapSizeUsable	87
6.12 StaticConstructorAttribute Class Reference	87
6.12.1 Detailed Description	88
6.13 StaticFieldAttribute Class Reference	88
6.13.1 Detailed Description	88
6.13.2 Constructor & Destructor Documentation	88
6.13.2.1 StaticFieldAttribute() [1/2]	88
6.13.2.2 StaticFieldAttribute() [2/2]	89
6.13.3 Property Documentation	89
6.13.3.1 Reset	89
6.14 StaticFieldResetMethodAttribute Class Reference	89
6.14.1 Detailed Description	89
6.14.2 Constructor & Destructor Documentation	89
6.14.2.1 StaticFieldResetMethodAttribute() [1/2]	89
6.14.2.2 StaticFieldResetMethodAttribute() [2/2]	90
6.15 Angle Struct Reference	90
6.15.1 Detailed Description	91
6.15.2 Member Function Documentation	91
6.15.2.1 Clamp() [1/2]	92
6.15.2.2 Clamp() [2/2]	92
6.15.2.3 Equals() [1/2]	92
6.15.2.4 Equals() [2/2]	92
6.15.2.5 GetHashCode()	93
6.15.2.6 Lerp()	93
6.15.2.7 Max()	93
6.15.2.8 Min()	93
6.15.2.9 operator Angle() [1/3]	93
6.15.2.10 operator Angle() [2/3]	94
6.15.2.11 operator Angle() [3/3]	94
6.15.2.12 operator double()	94
6.15.2.13 operator float()	96
6.15.2.14 operator"!=()	96
6.15.2.15 operator+()	97
6.15.2.16 operator-()	97
6.15.2.17 operator<()	97
6.15.2.18 operator<=()	98
6.15.2.19 operator==()	98
6.15.2.20 operator>()	99
6.15.2.21 operator>=()	99
6.15.2.22 ToString()	99

6.15.3 Member Data Documentation	99
6.15.3.1 SIZE	100
6.16 ArrayLengthAttribute Class Reference	100
6.16.1 Detailed Description	100
6.16.2 Constructor & Destructor Documentation	100
6.16.2.1 ArrayLengthAttribute() [1/2]	100
6.16.2.2 ArrayLengthAttribute() [2/2]	101
6.16.3 Property Documentation	101
6.16.3.1 MaxLength	101
6.16.3.2 MinLength	101
6.17 AssemblyNameAttribute Class Reference	101
6.17.1 Detailed Description	102
6.17.2 Property Documentation	102
6.17.2.1 RequiresUnsafeCode	102
6.18 AssetObject Class Reference	102
6.18.1 Detailed Description	102
6.19 TaskManager Class Reference	102
6.19.1 Detailed Description	103
6.19.2 Member Function Documentation	103
6.19.2.1 ContinueWhenAll()	103
6.19.2.2 Delay()	103
6.19.2.3 Run()	104
6.19.2.4 Service() [1/2]	104
6.19.2.5 Service() [2/2]	104
6.19.2.6 Setup()	106
6.20 AtomicInt Struct Reference	106
6.20.1 Detailed Description	107
6.20.2 Constructor & Destructor Documentation	107
6.20.2.1 AtomicInt()	107
6.20.3 Member Function Documentation	107
6.20.3.1 CompareExchange()	107
6.20.3.2 Decrement()	107
6.20.3.3 Exchange()	108
6.20.3.4 IncrementPost()	108
6.20.3.5 IncrementPre()	108
6.20.4 Member Data Documentation	108
6.20.4.1 _value	109
6.20.5 Property Documentation	109
6.20.5.1 Value	109
6.21 AuthorityMasks Class Reference	109
6.21.1 Detailed Description	109
6.21.2 Member Data Documentation	109

6.21.2.1 ALL	110
6.21.2.2 INPUT	110
6.21.2.3 NONE	110
6.21.2.4 PROXY	110
6.21.2.5 STATE	110
6.22 Behaviour Class Reference	110
6.22.1 Detailed Description	111
6.22.2 Member Function Documentation	111
6.22.2.1 AddBehaviour< T >()	111
6.22.2.2 DestroyBehaviour()	111
6.22.2.3 GetBehaviour< T >()	112
6.22.2.4 TryGetBehaviour< T >()	112
6.23 BinaryDataAttribute Class Reference	112
6.23.1 Detailed Description	112
6.24 BinUtils Class Reference	112
6.24.1 Detailed Description	113
6.24.2 Member Function Documentation	113
6.24.2.1 BytesToHex() [1/2]	113
6.24.2.2 BytesToHex() [2/2]	114
6.24.2.3 ByteToHex()	114
6.24.2.4 HexToBytes()	114
6.24.2.5 unsafe()	115
6.24.2.6 WordsToHex() [1/2]	115
6.24.2.7 WordsToHex() [2/2]	116
6.25 BitSetAttribute Class Reference	116
6.25.1 Detailed Description	116
6.25.2 Constructor & Destructor Documentation	117
6.25.2.1 BitSetAttribute()	117
6.25.3 Property Documentation	117
6.25.3.1 BitCount	117
6.26 CapacityAttribute Class Reference	117
6.26.1 Detailed Description	117
6.26.2 Constructor & Destructor Documentation	118
6.26.2.1 CapacityAttribute()	118
6.26.3 Property Documentation	118
6.26.3.1 Length	118
6.27 CRC64 Class Reference	118
6.27.1 Detailed Description	118
6.27.2 Member Function Documentation	118
6.27.2.1 Compute() [1/2]	119
6.27.2.2 Compute() [2/2]	120
6.28 DecoratingPropertyAttribute Class Reference	120

6.28.1 Detailed Description	121
6.28.2 Constructor & Destructor Documentation	121
6.28.2.1 DecoratingPropertyAttribute() [1/2]	121
6.28.2.2 DecoratingPropertyAttribute() [2/2]	121
6.28.3 Member Data Documentation	121
6.28.3.1 DefaultOrder	121
6.29 DefaultForPropertyAttribute Class Reference	122
6.29.1 Detailed Description	122
6.29.2 Constructor & Destructor Documentation	122
6.29.2.1 DefaultForPropertyAttribute()	122
6.29.3 Property Documentation	123
6.29.3.1 PropertyName	123
6.29.3.2 WordCount	123
6.29.3.3 WordOffset	123
6.30 DisplayAsEnumAttribute Class Reference	123
6.30.1 Detailed Description	124
6.30.2 Constructor & Destructor Documentation	124
6.30.2.1 DisplayAsEnumAttribute() [1/2]	124
6.30.2.2 DisplayAsEnumAttribute() [2/2]	124
6.30.3 Property Documentation	124
6.30.3.1 EnumType	124
6.30.3.2 EnumTypeMemberName	125
6.31 DisplayNameAttribute Class Reference	125
6.31.1 Detailed Description	125
6.31.2 Constructor & Destructor Documentation	125
6.31.2.1 DisplayNameAttribute()	125
6.31.3 Member Data Documentation	126
6.31.3.1 Name	126
6.32 DolfAttributeBase Class Reference	126
6.32.1 Detailed Description	127
6.32.2 Constructor & Destructor Documentation	127
6.32.2.1 DolfAttributeBase() [1/3]	127
6.32.2.2 DolfAttributeBase() [2/3]	127
6.32.2.3 DolfAttributeBase() [3/3]	128
6.32.3 Member Data Documentation	128
6.32.3.1 _doubleValue	128
6.32.3.2 _isDouble	128
6.32.3.3 _longValue	128
6.32.3.4 Compare	128
6.32.3.5 ConditionMember	129
6.32.3.6 ErrorOnConditionMemberNotFound	129
6.33 DrawerPropertyAttribute Class Reference	129

6.33.1 Detailed Description	129
6.34 DrawIfAttribute Class Reference	129
6.34.1 Detailed Description	130
6.34.2 Constructor & Destructor Documentation	130
6.34.2.1 DrawIfAttribute()	130
6.34.3 Member Data Documentation	130
6.34.3.1 Mode	130
6.34.4 Property Documentation	131
6.34.4.1 Hide	131
6.35 DrawInlineAttribute Class Reference	131
6.35.1 Detailed Description	131
6.36 DynamicHeap Struct Reference	131
6.36.1 Detailed Description	133
6.36.2 Member Function Documentation	133
6.36.2.1 CollectGarbage()	133
6.36.2.2 CollectGarbageDelegate()	133
6.36.2.3 Free()	133
6.36.2.4 SetForcedAlive< T >()	134
6.36.3 Member Data Documentation	134
6.36.3.1 _debruijnTable	134
6.37 DynamicHeap.Ignore Class Reference	135
6.37.1 Detailed Description	135
6.38 DynamicHeap.Instance Class Reference	135
6.38.1 Detailed Description	135
6.38.2 Constructor & Destructor Documentation	135
6.38.2.1 DynamicHeap.Instance()	135
6.38.3 Member Function Documentation	136
6.38.3.1 Allocate()	136
6.38.3.2 AllocateArray< T >()	136
6.38.3.3 AllocateArrayPointers< T >()	137
6.38.3.4 AllocateTracked< T >()	137
6.38.3.5 AllocateTrackedArray< T >()	138
6.38.3.6 AllocateTrackedArrayPointers< T >()	138
6.38.3.7 Free()	139
6.39 EditorButtonAttribute Class Reference	139
6.39.1 Detailed Description	140
6.39.2 Constructor & Destructor Documentation	140
6.39.2.1 EditorButtonAttribute() [1/2]	140
6.39.2.2 EditorButtonAttribute() [2/2]	140
6.39.3 Member Data Documentation	140
6.39.3.1 AllowMultipleTargets	141
6.39.3.2 DirtyObject	141

6.39.3.3 Label	141
6.39.3.4 Priority	141
6.39.3.5 Visibility	141
6.40 DataEncryptor Class Reference	141
6.40.1 Detailed Description	142
6.40.2 Member Function Documentation	142
6.40.2.1 Dispose()	142
6.40.2.2 EncryptData()	142
6.41 IDataEncryption Interface Reference	143
6.41.1 Detailed Description	143
6.41.2 Member Function Documentation	143
6.41.2.1 ComputeHash()	143
6.41.2.2 DecryptData()	144
6.41.2.3 EncryptData()	144
6.41.2.4 GenerateKey()	145
6.41.2.5 Setup()	145
6.41.2.6 VerifyHash()	145
6.42 EncryptionConfig Class Reference	146
6.42.1 Detailed Description	146
6.42.2 Member Data Documentation	146
6.42.2.1 EnableEncryption	146
6.43 EngineProfiler Class Reference	146
6.43.1 Detailed Description	148
6.43.2 Member Function Documentation	148
6.43.2.1 Begin()	148
6.43.2.2 End()	149
6.43.2.3 InputQueue()	149
6.43.2.4 InputRecvDelta()	149
6.43.2.5 InputRecvDeltaDeviation()	149
6.43.2.6 InputSize()	150
6.43.2.7 InterpolationOffset()	150
6.43.2.8 InterpolationOffsetDeviation()	150
6.43.2.9 InterpolationSpeed()	150
6.43.2.10 Resimulations()	151
6.43.2.11 RoundTripTime()	151
6.43.2.12 RpcIn()	151
6.43.2.13 RpcOut()	152
6.43.2.14 SimulationOffset()	152
6.43.2.15 SimulationOffsetDeviation()	152
6.43.2.16 SimulationSpeed()	152
6.43.2.17 StateRecvDelta()	153
6.43.2.18 StateRecvDeltaDeviation()	153

6.43.2.19 WorldSnapshotSize()	153
6.43.3 Member Data Documentation	153
6.43.3.1 InputQueueCallback	154
6.43.3.2 InputRecvDeltaCallback	154
6.43.3.3 InputRecvDeltaDeviationCallback	154
6.43.3.4 InputSizeCallback	154
6.43.3.5 InterpolationOffsetCallback	154
6.43.3.6 InterpolationOffsetDeviationCallback	154
6.43.3.7 InterpolationSpeedCallback	155
6.43.3.8 ResimulationsCallback	155
6.43.3.9 RoundTripTimeCallback	155
6.43.3.10 RpcInCallback	155
6.43.3.11 RpcOutCallback	155
6.43.3.12 SimulationOffsetCallback	155
6.43.3.13 SimulationOffsetDeviationCallback	156
6.43.3.14 SimulationSpeedCallback	156
6.43.3.15 StateRecvDeltaCallback	156
6.43.3.16 StateRecvDeltaDeviationCallback	156
6.43.3.17 WorldSnapshotSizeCallback	156
6.44 ErrorIfAttribute Class Reference	156
6.44.1 Detailed Description	157
6.44.2 Member Data Documentation	157
6.44.2.1 AsBox	157
6.44.2.2 Message	157
6.45 ExpandableEnumAttribute Class Reference	157
6.45.1 Detailed Description	158
6.45.2 Property Documentation	158
6.45.2.1 AlwaysExpanded	158
6.45.2.2 ShowFlagsButtons	158
6.45.2.3 ShowInlineHelp	158
6.46 FieldEditorButtonAttribute Class Reference	158
6.46.1 Detailed Description	159
6.46.2 Constructor & Destructor Documentation	159
6.46.2.1 FieldEditorButtonAttribute()	159
6.46.3 Member Data Documentation	159
6.46.3.1 AllowMultipleTargets	159
6.46.3.2 Label	160
6.46.3.3 TargetMethod	160
6.47 FieldsMask< T > Class Template Reference	160
6.47.1 Detailed Description	161
6.47.2 Constructor & Destructor Documentation	161
6.47.2.1 FieldsMask() [1/7]	161

6.47.2.2 FieldsMask() [2/7]	161
6.47.2.3 FieldsMask() [3/7]	161
6.47.2.4 FieldsMask() [4/7]	162
6.47.2.5 FieldsMask() [5/7]	162
6.47.2.6 FieldsMask() [6/7]	162
6.47.2.7 FieldsMask() [7/7]	162
6.47.3 Member Function Documentation	162
6.47.3.1 operator Mask256()	162
6.47.4 Member Data Documentation	163
6.47.4.1 Mask	163
6.48 FixedArray< T > Class Template Reference	163
6.48.1 Detailed Description	164
6.48.2 Constructor & Destructor Documentation	164
6.48.2.1 FixedArray()	165
6.48.3 Member Function Documentation	165
6.48.3.1 Clear()	165
6.48.3.2 CopyFrom() [1/2]	165
6.48.3.3 CopyFrom() [2/2]	165
6.48.3.4 CopyTo() [1/2]	166
6.48.3.5 CopyTo() [2/2]	166
6.48.3.6 Create< T >()	167
6.48.3.7 Create< TActual, TAdapted >()	167
6.48.3.8 CreateFromFieldSequence< T >()	168
6.48.3.9 GetEnumerator()	168
6.48.3.10 IndexOf< T >()	168
6.48.3.11 ToArray()	169
6.48.3.12 ToListString()	169
6.48.3.13 ToString()	169
6.48.4 Property Documentation	169
6.48.4.1 Length	169
6.48.4.2 this[int index]	169
6.49 FixedArray< T >.Enumerator Struct Reference	170
6.49.1 Detailed Description	170
6.49.2 Constructor & Destructor Documentation	170
6.49.2.1 Enumerator()	170
6.49.3 Member Function Documentation	171
6.49.3.1 Dispose()	171
6.49.3.2 MoveNext()	171
6.49.3.3 Reset()	171
6.49.4 Property Documentation	171
6.49.4.1 Current	171
6.50 FixedBufferPropertyAttribute Class Reference	172

6.50.1 Detailed Description	172
6.50.2 Constructor & Destructor Documentation	172
6.50.2.1 FixedBufferPropertyAttribute()	172
6.50.3 Property Documentation	173
6.50.3.1 Capacity	173
6.50.3.2 SurrogateType	173
6.50.3.3 Type	173
6.51 FixedStorage Class Reference	173
6.51.1 Detailed Description	173
6.51.2 Member Function Documentation	173
6.51.2.1 GetWordCount< T >()	173
6.52 FloatCompressed Struct Reference	174
6.52.1 Detailed Description	174
6.52.2 Member Function Documentation	175
6.52.2.1 Equals() [1/2]	175
6.52.2.2 Equals() [2/2]	175
6.52.2.3 GetHashCode()	175
6.52.2.4 operator float()	176
6.52.2.5 operator FloatCompressed()	176
6.52.2.6 operator"!="()	176
6.52.2.7 operator==()	177
6.52.3 Member Data Documentation	177
6.52.3.1 valueEncoded	177
6.53 FloatUtils Class Reference	177
6.53.1 Detailed Description	178
6.53.2 Member Function Documentation	178
6.53.2.1 Compress()	178
6.53.2.2 Decompress()	178
6.53.3 Member Data Documentation	179
6.53.3.1 DEFAULT_ACCURACY	179
6.54 FusionGlobalScriptableObject< T > Class Template Reference	179
6.54.1 Detailed Description	180
6.54.2 Member Function Documentation	180
6.54.2.1 OnDisable()	180
6.54.2.2 OnLoadedAsGlobal()	180
6.54.2.3 OnUnloadedAsGlobal()	180
6.54.2.4 TryGetGlobalInternal()	180
6.54.2.5 UnloadGlobalInternal()	181
6.54.3 Property Documentation	181
6.54.3.1 GlobalInternal	181
6.54.3.2 IsGlobal	181
6.54.3.3 IsGlobalLoadedInternal	182

6.55 FusionGlobalScriptableObjectAttribute Class Reference	182
6.55.1 Detailed Description	182
6.55.2 Constructor & Destructor Documentation	182
6.55.2.1 FusionGlobalScriptableObjectAttribute()	182
6.55.3 Property Documentation	183
6.55.3.1 DefaultContents	183
6.55.3.2 DefaultContentsGeneratorMethod	183
6.55.3.3 DefaultPath	183
6.56 FusionGlobalScriptableObjectLoadResult Struct Reference	183
6.56.1 Detailed Description	184
6.56.2 Constructor & Destructor Documentation	184
6.56.2.1 FusionGlobalScriptableObjectLoadResult()	184
6.56.3 Member Function Documentation	184
6.56.3.1 operator FusionGlobalScriptableObjectLoadResult()	184
6.56.4 Member Data Documentation	184
6.56.4.1 Object	184
6.56.4.2 Unloader	185
6.57 FusionGlobalScriptableObjectSourceAttribute Class Reference	185
6.57.1 Detailed Description	185
6.57.2 Member Function Documentation	186
6.57.2.1 Load()	186
6.57.3 Property Documentation	186
6.57.3.1 AllowEditMode	186
6.57.3.2 AllowFallback	186
6.57.3.3 ObjectType	186
6.57.3.4 Order	186
6.58 FusionMonoBehaviour Class Reference	187
6.58.1 Detailed Description	187
6.59 FusionScriptableObject Class Reference	187
6.59.1 Detailed Description	187
6.60 HeapConfiguration Class Reference	187
6.60.1 Detailed Description	188
6.60.2 Member Function Documentation	188
6.60.2.1 Init()	188
6.60.2.2 ToString()	188
6.60.3 Member Data Documentation	188
6.60.3.1 GlobalsSize	188
6.60.3.2 PageCount	188
6.60.3.3 PageShift	189
6.61 HideArrayElementLabelAttribute Class Reference	189
6.61.1 Detailed Description	189
6.61.2 Constructor & Destructor Documentation	189

6.61.2.1 HideArrayElementLabelAttribute()	189
6.62 Hitbox Class Reference	189
6.62.1 Detailed Description	190
6.62.2 Member Function Documentation	191
6.62.2.1 DrawGizmos()	191
6.62.2.2 OnDrawGizmos()	191
6.62.2.3 SetLayer()	191
6.62.3 Member Data Documentation	191
6.62.3.1 BoxExtents	191
6.62.3.2 CapsuleExtents	192
6.62.3.3 CapsuleRadius	192
6.62.3.4 GizmosColor	192
6.62.3.5 Offset	192
6.62.3.6 Root	192
6.62.3.7 SphereRadius	192
6.62.3.8 Type	193
6.62.4 Property Documentation	193
6.62.4.1 ColliderIndex	193
6.62.4.2 HitboxActive	193
6.62.4.3 HitboxIndex	193
6.62.4.4 Position	193
6.63 HitboxManager Class Reference	193
6.63.1 Detailed Description	195
6.63.2 Member Function Documentation	196
6.63.2.1 GetPlayerTickAndAlpha()	196
6.63.2.2 GetStatisticsSnapshot()	196
6.63.2.3 OverlapBox() [1/3]	196
6.63.2.4 OverlapBox() [2/3]	197
6.63.2.5 OverlapBox() [3/3]	198
6.63.2.6 OverlapSphere() [1/3]	199
6.63.2.7 OverlapSphere() [2/3]	199
6.63.2.8 OverlapSphere() [3/3]	200
6.63.2.9 PositionRotation() [1/2]	201
6.63.2.10 PositionRotation() [2/2]	201
6.63.2.11 Raycast() [1/3]	202
6.63.2.12 Raycast() [2/3]	202
6.63.2.13 Raycast() [3/3]	203
6.63.2.14 RaycastAll() [1/3]	204
6.63.2.15 RaycastAll() [2/3]	205
6.63.2.16 RaycastAll() [3/3]	206
6.63.3 Member Data Documentation	206
6.63.3.1 BVHDepth	207

6.63.3.2 BVHNodes	207
6.63.3.3 DrawInfo	207
6.63.3.4 TotalHitboxes	207
6.64 HitboxRoot Class Reference	207
6.64.1 Detailed Description	209
6.64.2 Member Enumeration Documentation	209
6.64.2.1 ConfigFlags	209
6.64.3 Member Function Documentation	209
6.64.3.1 DrawGizmos()	209
6.64.3.2 InitHitboxes()	210
6.64.3.3 IsHitboxActive()	210
6.64.3.4 OnDrawGizmos()	210
6.64.3.5 SetHitboxActive()	210
6.64.3.6 SetMinBoundingRadius()	211
6.64.4 Member Data Documentation	211
6.64.4.1 BroadRadius	211
6.64.4.2 Config	211
6.64.4.3 GizmosColor	212
6.64.4.4 Hitboxes	212
6.64.4.5 MAX_HITBOXES	212
6.64.4.6 Offset	212
6.64.5 Property Documentation	212
6.64.5.1 HitboxRootActive	212
6.64.5.2 InInterest	213
6.64.5.3 Manager	213
6.65 HostMigrationConfig Class Reference	213
6.65.1 Detailed Description	213
6.65.2 Member Data Documentation	213
6.65.2.1 EnableAutoUpdate	213
6.65.2.2 UpdateDelay	213
6.66 HostMigrationToken Class Reference	214
6.66.1 Detailed Description	214
6.66.2 Property Documentation	214
6.66.2.1 GameMode	214
6.67 IAfterAllTicks Interface Reference	214
6.67.1 Detailed Description	214
6.67.2 Member Function Documentation	215
6.67.2.1 AfterAllTicks()	215
6.68 IAfterClientPredictionReset Interface Reference	216
6.68.1 Detailed Description	216
6.68.2 Member Function Documentation	216
6.68.2.1 AfterClientPredictionReset()	216

6.69 IAfterHostMigration Interface Reference	216
6.69.1 Detailed Description	217
6.69.2 Member Function Documentation	217
6.69.2.1 AfterHostMigration()	217
6.70 IAfterRender Interface Reference	217
6.70.1 Detailed Description	217
6.70.2 Member Function Documentation	217
6.70.2.1 AfterRender()	217
6.71 IAfterSpawned Interface Reference	218
6.71.1 Detailed Description	218
6.71.2 Member Function Documentation	218
6.71.2.1 AfterSpawned()	218
6.72 IAfterTick Interface Reference	218
6.72.1 Detailed Description	218
6.72.2 Member Function Documentation	219
6.72.2.1 AfterTick()	219
6.73 IAfterUpdate Interface Reference	219
6.73.1 Detailed Description	219
6.73.2 Member Function Documentation	219
6.73.2.1 AfterUpdate()	219
6.74 IAfterUpdateRemotePrefabs Interface Reference	219
6.74.1 Detailed Description	220
6.74.2 Member Function Documentation	220
6.74.2.1 AfterUpdateRemotePrefabs()	220
6.75 IAsyncOperation Interface Reference	220
6.75.1 Detailed Description	220
6.75.2 Property Documentation	221
6.75.2.1 Error	221
6.75.2.2 IsDone	221
6.75.3 Event Documentation	221
6.75.3.1 Completed	221
6.76 IBeforeAllTicks Interface Reference	221
6.76.1 Detailed Description	222
6.76.2 Member Function Documentation	222
6.76.2.1 BeforeAllTicks()	222
6.77 IBeforeClientPredictionReset Interface Reference	222
6.77.1 Detailed Description	222
6.77.2 Member Function Documentation	222
6.77.2.1 BeforeClientPredictionReset()	223
6.78 IBeforeCopyPreviousState Interface Reference	223
6.78.1 Detailed Description	223
6.78.2 Member Function Documentation	223

6.78.2.1 BeforeCopyPreviousState()	223
6.79 IBeforeHitboxRegistration Interface Reference	223
6.79.1 Detailed Description	224
6.79.2 Member Function Documentation	224
6.79.2.1 BeforeHitboxRegistration()	224
6.80 IBeforeSimulation Interface Reference	224
6.80.1 Detailed Description	224
6.80.2 Member Function Documentation	224
6.80.2.1 BeforeSimulation()	224
6.81 IBeforeTick Interface Reference	225
6.81.1 Detailed Description	225
6.81.2 Member Function Documentation	225
6.81.2.1 BeforeTick()	225
6.82 IBeforeUpdate Interface Reference	225
6.82.1 Detailed Description	226
6.82.2 Member Function Documentation	226
6.82.2.1 BeforeUpdate()	226
6.83 IBeforeUpdateRemotePrefabs Interface Reference	226
6.83.1 Detailed Description	226
6.83.2 Member Function Documentation	226
6.83.2.1 BeforeUpdateRemotePrefabs()	226
6.84 ICoroutine Interface Reference	227
6.84.1 Detailed Description	227
6.85 IDespawned Interface Reference	227
6.85.1 Detailed Description	227
6.85.2 Member Function Documentation	227
6.85.2.1 Despawned()	227
6.86 IElementReaderWriter< T > Interface Template Reference	228
6.86.1 Detailed Description	228
6.86.2 Member Function Documentation	228
6.86.2.1 GetElementHashCode()	228
6.86.2.2 GetElementWordCount()	229
6.86.2.3 Read()	229
6.86.2.4 ReadRef()	229
6.86.2.5 Write()	230
6.87 IFixedStorage Interface Reference	230
6.87.1 Detailed Description	230
6.88 IInputAuthorityGained Interface Reference	230
6.88.1 Detailed Description	230
6.88.2 Member Function Documentation	231
6.88.2.1 InputAuthorityGained()	231
6.89 IInputAuthorityLost Interface Reference	231

6.89.1 Detailed Description	231
6.89.2 Member Function Documentation	231
6.89.2.1 InputAuthorityLost()	231
6.90 IInterestEnter Interface Reference	231
6.90.1 Detailed Description	232
6.90.2 Member Function Documentation	232
6.90.2.1 InterestEnter()	232
6.91 IInterestExit Interface Reference	232
6.91.1 Detailed Description	232
6.91.2 Member Function Documentation	232
6.91.2.1 InterestExit()	233
6.92 ILocalPrefabCreated Interface Reference	233
6.92.1 Detailed Description	233
6.92.2 Member Function Documentation	233
6.92.2.1 LocalPrefabCreated()	233
6.93 INetworkArray Interface Reference	233
6.93.1 Detailed Description	234
6.93.2 Property Documentation	234
6.93.2.1 this[int index]	234
6.94 INetworkAssetSource< T > Interface Template Reference	234
6.94.1 Detailed Description	234
6.94.2 Member Function Documentation	235
6.94.2.1 Acquire()	235
6.94.2.2 Release()	235
6.94.2.3 WaitForResult()	235
6.94.3 Property Documentation	235
6.94.3.1 Description	236
6.94.3.2 IsCompleted	236
6.95 INetworkDictionary Interface Reference	236
6.95.1 Detailed Description	236
6.95.2 Member Function Documentation	236
6.95.2.1 Add()	236
6.96 INetworkInput Interface Reference	237
6.96.1 Detailed Description	237
6.97 INetworkLinkedList Interface Reference	237
6.97.1 Detailed Description	237
6.97.2 Member Function Documentation	237
6.97.2.1 Add()	237
6.98 INetworkObjectInitializer Interface Reference	238
6.98.1 Detailed Description	238
6.98.2 Member Function Documentation	238
6.98.2.1 InitializeNetworkState()	238

6.99 INetworkObjectProvider Interface Reference	238
6.99.1 Detailed Description	239
6.99.2 Member Function Documentation	239
6.99.2.1 AcquirePrefabInstance()	239
6.99.2.2 GetPrefabId()	240
6.99.2.3 Initialize()	240
6.99.2.4 ReleaseInstance()	240
6.99.2.5 Shutdown()	241
6.100 INetworkPrefabSource Interface Reference	241
6.100.1 Detailed Description	241
6.100.2 Property Documentation	241
6.100.2.1 AssetGuid	241
6.101 INetworkRunnerCallbacks Interface Reference	241
6.101.1 Detailed Description	242
6.101.2 Member Function Documentation	243
6.101.2.1 OnConnectedToServer()	243
6.101.2.2 OnConnectFailed()	243
6.101.2.3 OnConnectRequest()	243
6.101.2.4 OnCustomAuthenticationResponse()	243
6.101.2.5 OnDisconnectedFromServer()	244
6.101.2.6 OnHostMigration()	244
6.101.2.7 OnInput()	244
6.101.2.8 OnInputMissing()	244
6.101.2.9 OnObjectEnterAOI()	245
6.101.2.10 OnObjectExitAOI()	245
6.101.2.11 OnPlayerJoined()	245
6.101.2.12 OnPlayerLeft()	246
6.101.2.13 OnReliableDataProgress()	246
6.101.2.14 OnReliableDataReceived()	246
6.101.2.15 OnSceneLoadDone()	247
6.101.2.16 OnSceneLoadStart()	247
6.101.2.17 OnSessionListUpdated()	247
6.101.2.18 OnShutdown()	247
6.101.2.19 OnUserSimulationMessage()	248
6.102 INetworkRunnerUpdater Interface Reference	248
6.102.1 Detailed Description	248
6.102.2 Member Function Documentation	248
6.102.2.1 Initialize()	249
6.102.2.2 Shutdown()	249
6.103 INetworkSceneManager Interface Reference	249
6.103.1 Detailed Description	250
6.103.2 Member Function Documentation	250

6.103.2.1 GetSceneRef() [1/2]	250
6.103.2.2 GetSceneRef() [2/2]	250
6.103.2.3 Initialize()	251
6.103.2.4 IsRunnerScene()	251
6.103.2.5 LoadScene()	251
6.103.2.6 MakeDontDestroyOnLoad()	251
6.103.2.7 MoveGameObjectToScene()	251
6.103.2.8 OnSceneInfoChanged()	252
6.103.2.9 Shutdown()	252
6.103.2.10 TryGetPhysicsScene2D()	252
6.103.2.11 TryGetPhysicsScene3D()	252
6.103.2.12 UnloadScene()	253
6.103.3 Property Documentation	253
6.103.3.1 IsBusy	253
6.103.3.2 MainRunnerScene	253
6.104 INetworkStruct Interface Reference	253
6.104.1 Detailed Description	253
6.105 INetworkTRSPTeleport Interface Reference	254
6.105.1 Detailed Description	254
6.105.2 Member Function Documentation	254
6.105.2.1 Teleport()	254
6.106 InlineHelpAttribute Class Reference	254
6.106.1 Detailed Description	255
6.106.2 Constructor & Destructor Documentation	255
6.106.2.1 InlineHelpAttribute()	255
6.106.3 Property Documentation	255
6.106.3.1 ShowTypeHelp	255
6.107 IUnitySurrogate Interface Reference	255
6.107.1 Detailed Description	255
6.107.2 Member Function Documentation	255
6.107.2.1 Read()	255
6.107.2.2 Write()	256
6.108 IUnityValueSurrogate< T > Interface Template Reference	256
6.108.1 Detailed Description	256
6.108.2 Property Documentation	257
6.108.2.1 DataProperty	257
6.109 UnityArraySurrogate< T, ReaderWriter > Class Template Reference	257
6.109.1 Detailed Description	257
6.109.2 Member Function Documentation	258
6.109.2.1 Init()	258
6.109.2.2 Read()	258
6.109.2.3 Write()	258

6.109.3 Property Documentation	259
6.109.3.1 DataProperty	259
6.110 UnityDictionarySurrogate< TKeyType, TKeyReaderWriter, TValueType, TValueReaderWriter > Class Template Reference	259
6.110.1 Detailed Description	259
6.110.2 Member Function Documentation	260
6.110.2.1 Init()	260
6.110.2.2 Read()	260
6.110.2.3 Write()	261
6.110.3 Property Documentation	261
6.110.3.1 DataProperty	261
6.111 UnityLinkedListSurrogate< T, ReaderWriter > Class Template Reference	261
6.111.1 Detailed Description	262
6.111.2 Member Function Documentation	262
6.111.2.1 Init()	262
6.111.2.2 Read()	263
6.111.2.3 Write()	263
6.111.3 Property Documentation	263
6.111.3.1 DataProperty	263
6.112 UnitySurrogateBase Class Reference	263
6.112.1 Detailed Description	264
6.112.2 Member Function Documentation	264
6.112.2.1 Init()	264
6.112.2.2 Read()	264
6.112.2.3 Write()	265
6.113 UnityValueSurrogate< T, TReaderWriter > Class Template Reference	265
6.113.1 Detailed Description	265
6.113.2 Member Function Documentation	266
6.113.2.1 Init()	266
6.113.2.2 Read()	266
6.113.2.3 Write()	267
6.113.3 Property Documentation	267
6.113.3.1 DataProperty	267
6.114 InterpolatedErrorCorrectionSettings Class Reference	267
6.114.1 Detailed Description	267
6.114.2 Member Data Documentation	268
6.114.2.1 MaxRate	268
6.114.2.2 MinRate	268
6.114.2.3 PosBlendEnd	268
6.114.2.4 PosBlendStart	269
6.114.2.5 PosMinCorrection	269
6.114.2.6 PosTeleportDistance	269

6.114.2.7 RotBlendEnd	269
6.114.2.8 RotBlendStart	270
6.114.2.9 RotTeleportRadians	270
6.115 IPlayerJoined Interface Reference	270
6.115.1 Detailed Description	270
6.115.2 Member Function Documentation	270
6.115.2.1 PlayerJoined()	270
6.116 IPlayerLeft Interface Reference	271
6.116.1 Detailed Description	271
6.116.2 Member Function Documentation	271
6.116.2.1 PlayerLeft()	271
6.117 IRemotePrefabCreated Interface Reference	271
6.117.1 Detailed Description	272
6.117.2 Member Function Documentation	272
6.117.2.1 RemotePrefabCreated()	272
6.118 ISceneLoadDone Interface Reference	272
6.118.1 Detailed Description	272
6.118.2 Member Function Documentation	272
6.118.2.1 SceneLoadDone()	272
6.119 ISceneLoadStart Interface Reference	273
6.119.1 Detailed Description	273
6.119.2 Member Function Documentation	273
6.119.2.1 SceneLoadStart()	273
6.120 ISimulationEnter Interface Reference	273
6.120.1 Detailed Description	274
6.120.2 Member Function Documentation	274
6.120.2.1 SimulationEnter()	274
6.121 ISimulationExit Interface Reference	274
6.121.1 Detailed Description	274
6.121.2 Member Function Documentation	274
6.121.2.1 SimulationExit()	275
6.122 ISpawned Interface Reference	275
6.122.1 Detailed Description	275
6.122.2 Member Function Documentation	275
6.122.2.1 Spawned()	275
6.123 IStateAuthorityChanged Interface Reference	275
6.123.1 Detailed Description	276
6.123.2 Member Function Documentation	276
6.123.2.1 StateAuthorityChanged()	276
6.124 LagCompensatedHit Struct Reference	276
6.124.1 Detailed Description	277
6.124.2 Member Function Documentation	277

6.124.2.1 operator LagCompensatedHit() [1/2]	277
6.124.2.2 operator LagCompensatedHit() [2/2]	277
6.124.3 Member Data Documentation	277
6.124.3.1 Collider	278
6.124.3.2 Collider2D	278
6.124.3.3 Distance	278
6.124.3.4 GameObject	278
6.124.3.5 Hitbox	278
6.124.3.6 HitboxColliderPosition	278
6.124.3.7 HitboxColliderRotation	279
6.124.3.8 Normal	279
6.124.3.9 Point	279
6.124.3.10 Type	279
6.125 AABB Struct Reference	279
6.125.1 Detailed Description	280
6.125.2 Constructor & Destructor Documentation	280
6.125.2.1 AABB() [1/3]	280
6.125.2.2 AABB() [2/3]	280
6.125.2.3 AABB() [3/3]	281
6.125.3 Member Data Documentation	281
6.125.3.1 Center	281
6.125.3.2 Extents	281
6.125.3.3 Max	281
6.125.3.4 Min	281
6.126 BoxOverlapQuery Class Reference	281
6.126.1 Detailed Description	282
6.126.2 Constructor & Destructor Documentation	282
6.126.2.1 BoxOverlapQuery() [1/2]	282
6.126.2.2 BoxOverlapQuery() [2/2]	283
6.126.3 Member Function Documentation	283
6.126.3.1 Check()	283
6.126.4 Member Data Documentation	283
6.126.4.1 Center	283
6.126.4.2 Extents	284
6.126.4.3 Rotation	284
6.127 BoxOverlapQueryParams Struct Reference	284
6.127.1 Detailed Description	284
6.127.2 Constructor & Destructor Documentation	284
6.127.2.1 BoxOverlapQueryParams()	284
6.127.3 Member Data Documentation	285
6.127.3.1 Center	285
6.127.3.2 Extents	285

6.127.3.3 QueryParams	285
6.127.3.4 Rotation	285
6.127.3.5 StaticHitsCapacity	286
6.128 BVHDraw Class Reference	286
6.128.1 Detailed Description	286
6.128.2 Member Function Documentation	286
6.128.2.1 GetEnumerator()	286
6.129 BVHNodeDrawInfo Class Reference	286
6.129.1 Detailed Description	287
6.129.2 Property Documentation	287
6.129.2.1 Bounds	287
6.129.2.2 Depth	287
6.129.2.3 MaxDepth	287
6.130 ColliderDrawInfo Class Reference	287
6.130.1 Detailed Description	288
6.130.2 Property Documentation	288
6.130.2.1 BoxExtents	288
6.130.2.2 CapsuleBottomCenter	288
6.130.2.3 CapsuleExtents	288
6.130.2.4 CapsuleTopCenter	288
6.130.2.5 LocalToWorldMatrix	289
6.130.2.6 Offset	289
6.130.2.7 Radius	289
6.130.2.8 Type	289
6.131 HitboxColliderContainerDraw Class Reference	289
6.131.1 Detailed Description	289
6.131.2 Member Function Documentation	290
6.131.2.1 GetEnumerator()	290
6.132 LagCompensatedExt Class Reference	290
6.132.1 Detailed Description	290
6.132.2 Member Function Documentation	290
6.132.2.1 SortDistance()	290
6.132.2.2 SortReference()	291
6.133 LagCompensationDraw Class Reference	291
6.133.1 Detailed Description	291
6.133.2 Member Function Documentation	291
6.133.2.1 GizmosDrawWireCapsule()	292
6.133.3 Member Data Documentation	292
6.133.3.1 BVHDraw	292
6.133.3.2 SnapshotHistoryDraw	292
6.134 LagCompensationUtils.ContactData Struct Reference	292
6.134.1 Detailed Description	293

6.134.2 Member Data Documentation	293
6.134.2.1 Normal	293
6.134.2.2 Penetration	293
6.134.2.3 Point	293
6.135 PositionRotationQueryParams Struct Reference	293
6.135.1 Detailed Description	294
6.135.2 Constructor & Destructor Documentation	294
6.135.2.1 PositionRotationQueryParams()	294
6.135.3 Member Data Documentation	294
6.135.3.1 Hitbox	294
6.135.3.2 QueryParams	294
6.136 Query Class Reference	294
6.136.1 Detailed Description	295
6.136.2 Constructor & Destructor Documentation	295
6.136.2.1 Query()	295
6.136.3 Member Function Documentation	296
6.136.3.1 Check()	296
6.136.4 Member Data Documentation	296
6.136.4.1 Alpha	296
6.136.4.2 LayerMask	296
6.136.4.3 Options	296
6.136.4.4 Player	297
6.136.4.5 PreProcessingDelegate	297
6.136.4.6 Tick	297
6.136.4.7 TickTo	297
6.136.4.8 TriggerInteraction	297
6.136.4.9 UserArgs	297
6.137 QueryParams Struct Reference	298
6.137.1 Detailed Description	298
6.137.2 Member Data Documentation	298
6.137.2.1 Alpha	298
6.137.2.2 LayerMask	298
6.137.2.3 Options	299
6.137.2.4 Player	299
6.137.2.5 PreProcessingDelegate	299
6.137.2.6 Tick	299
6.137.2.7 TickTo	299
6.137.2.8 TriggerInteraction	299
6.137.2.9 UserArgs	300
6.138 RaycastAllQuery Class Reference	300
6.138.1 Detailed Description	300
6.138.2 Constructor & Destructor Documentation	300

6.138.2.1 RaycastAllQuery() [1/2]	300
6.138.2.2 RaycastAllQuery() [2/2]	301
6.139 RaycastQuery Class Reference	301
6.139.1 Detailed Description	301
6.139.2 Constructor & Destructor Documentation	302
6.139.2.1 RaycastQuery()	302
6.139.3 Member Function Documentation	302
6.139.3.1 Check()	302
6.139.4 Member Data Documentation	302
6.139.4.1 Direction	302
6.139.4.2 Length	303
6.139.4.3 Origin	303
6.140 RaycastQueryParams Struct Reference	303
6.140.1 Detailed Description	303
6.140.2 Constructor & Destructor Documentation	303
6.140.2.1 RaycastQueryParams()	303
6.140.3 Member Data Documentation	304
6.140.3.1 Direction	304
6.140.3.2 Length	304
6.140.3.3 Origin	304
6.140.3.4 QueryParams	304
6.140.3.5 StaticHitsCapacity	305
6.141 SnapshotHistoryDraw Class Reference	305
6.141.1 Detailed Description	305
6.141.2 Member Function Documentation	305
6.141.2.1 GetEnumerator()	305
6.142 SphereOverlapQuery Class Reference	305
6.142.1 Detailed Description	306
6.142.2 Constructor & Destructor Documentation	306
6.142.2.1 SphereOverlapQuery() [1/2]	306
6.142.2.2 SphereOverlapQuery() [2/2]	306
6.142.3 Member Function Documentation	307
6.142.3.1 Check()	307
6.142.4 Member Data Documentation	307
6.142.4.1 Center	307
6.142.4.2 Radius	307
6.143 SphereOverlapQueryParams Struct Reference	307
6.143.1 Detailed Description	308
6.143.2 Constructor & Destructor Documentation	308
6.143.2.1 SphereOverlapQueryParams()	308
6.143.3 Member Data Documentation	308
6.143.3.1 Center	309

6.143.3.2 QueryParams	309
6.143.3.3 Radius	309
6.143.3.4 StaticHitsCapacity	309
6.144 LagCompensationSettings Class Reference	309
6.144.1 Detailed Description	310
6.144.2 Member Data Documentation	310
6.144.2.1 CachedStaticCollidersSize	310
6.144.2.2 Enabled	310
6.144.2.3 HitboxBufferLengthInMs	310
6.144.2.4 HitboxDefaultCapacity	310
6.144.3 Property Documentation	311
6.144.3.1 ExpansionFactor	311
6.144.3.2 Optimize	311
6.145 LayerAttribute Class Reference	311
6.145.1 Detailed Description	311
6.146 LayerMatrixAttribute Class Reference	311
6.146.1 Detailed Description	311
6.147 LobbyInfo Class Reference	311
6.147.1 Detailed Description	312
6.147.2 Property Documentation	312
6.147.2.1 IsValid	312
6.147.2.2 Name	312
6.147.2.3 Region	312
6.148 Mask256 Struct Reference	312
6.148.1 Detailed Description	313
6.148.2 Constructor & Destructor Documentation	313
6.148.2.1 Mask256()	314
6.148.3 Member Function Documentation	314
6.148.3.1 Clear()	314
6.148.3.2 Equals() [1/2]	314
6.148.3.3 Equals() [2/2]	314
6.148.3.4 GetBit()	314
6.148.3.5 GetHashCode()	315
6.148.3.6 IsNothing()	315
6.148.3.7 operator long()	315
6.148.3.8 operator Mask256()	315
6.148.3.9 operator&()	315
6.148.3.10 operator" ()	316
6.148.3.11 operator~()	316
6.148.3.12 SetBit()	316
6.148.3.13 ToString()	316
6.148.4 Property Documentation	316

6.148.4.1 this[int i]	316
6.149 Maths Class Reference	317
6.149.1 Detailed Description	318
6.149.2 Member Function Documentation	318
6.149.2.1 BitScanReverse() [1/4]	319
6.149.2.2 BitScanReverse() [2/4]	319
6.149.2.3 BitScanReverse() [3/4]	319
6.149.2.4 BitScanReverse() [4/4]	320
6.149.2.5 BitsRequiredForNumber() [1/2]	320
6.149.2.6 BitsRequiredForNumber() [2/2]	320
6.149.2.7 BytesRequiredForBits() [1/2]	321
6.149.2.8 BytesRequiredForBits() [2/2]	321
6.149.2.9 CeilToInt()	321
6.149.2.10 Clamp() [1/4]	322
6.149.2.11 Clamp() [2/4]	322
6.149.2.12 Clamp() [3/4]	323
6.149.2.13 Clamp() [4/4]	323
6.149.2.14 Clamp01() [1/2]	324
6.149.2.15 Clamp01() [2/2]	324
6.149.2.16 ClampToByte()	324
6.149.2.17 CosineInterpolate()	325
6.149.2.18 CountSetBits()	325
6.149.2.19 CountUsedBitsMinOne()	325
6.149.2.20 FloorToInt()	326
6.149.2.21 IntsRequiredForBits()	326
6.149.2.22 Lerp() [1/2]	326
6.149.2.23 Lerp() [2/2]	327
6.149.2.24 MicrosecondsToSeconds()	327
6.149.2.25 MillisecondsToMicroseconds()	328
6.149.2.26 MillisecondsToSeconds()	328
6.149.2.27 Min()	328
6.149.2.28 NextPowerOfTwo()	329
6.149.2.29 PrintBits()	329
6.149.2.30 QuaternionCompress()	330
6.149.2.31 QuaternionDecompress()	330
6.149.2.32 SecondsToMicroseconds()	330
6.149.2.33 SecondsToMilliseconds()	331
6.149.2.34 SizeOfBits< T >()	331
6.149.2.35 ZigZagDecode() [1/2]	331
6.149.2.36 ZigZagDecode() [2/2]	332
6.149.2.37 ZigZagEncode() [1/2]	332
6.149.2.38 ZigZagEncode() [2/2]	332

6.150 Maths.FastAbs Struct Reference	334
6.150.1 Detailed Description	334
6.150.2 Member Data Documentation	334
6.150.2.1 Mask	334
6.150.2.2 Single	334
6.150.2.3 UInt32	335
6.151 MaxStringByteCountAttribute Class Reference	335
6.151.1 Detailed Description	335
6.151.2 Constructor & Destructor Documentation	335
6.151.2.1 MaxStringByteCountAttribute()	335
6.151.3 Property Documentation	336
6.151.3.1 ByteCount	336
6.151.3.2 Encoding	336
6.152 Native Class Reference	336
6.152.1 Detailed Description	339
6.152.2 Member Function Documentation	339
6.152.2.1 AlignPointer()	339
6.152.2.2 ArrayClear< T >()	340
6.152.2.3 ArrayCompare< T >()	340
6.152.2.4 ArrayCopy()	341
6.152.2.5 CopyFromArray< T >()	341
6.152.2.6 CopyToArray< T >()	342
6.152.2.7 DoubleArray< T >()	342
6.152.2.8 DoublePtrArray< T >()	343
6.152.2.9 Empty< T >()	343
6.152.2.10 Expand()	344
6.152.2.11 ExpandArray< T >()	344
6.152.2.12 ExpandPtrArray< T >()	345
6.152.2.13 Free()	345
6.152.2.14 GetAlignment()	345
6.152.2.15 GetAlignment< T >()	346
6.152.2.16 GetFieldOffset()	346
6.152.2.17 GetLengthPrefixedUTF8ByteCount()	347
6.152.2.18 GetMaxAlignment() [1/4]	347
6.152.2.19 GetMaxAlignment() [2/4]	347
6.152.2.20 GetMaxAlignment() [3/4]	348
6.152.2.21 GetMaxAlignment() [4/4]	348
6.152.2.22 IsAligned()	349
6.152.2.23 IsPointerAligned()	349
6.152.2.24 Malloc()	349
6.152.2.25 Malloc< T >()	350
6.152.2.26 MallocAndClear()	350

6.152.2.27 MallocAndClear< T >()	351
6.152.2.28 MallocAndClearArray()	351
6.152.2.29 MallocAndClearArray< T >()	351
6.152.2.30 MallocAndClearArrayMin1< T >()	352
6.152.2.31 MallocAndClearBlock() [1/11]	352
6.152.2.32 MallocAndClearBlock() [2/11]	353
6.152.2.33 MallocAndClearBlock() [3/11]	354
6.152.2.34 MallocAndClearBlock() [4/11]	354
6.152.2.35 MallocAndClearBlock() [5/11]	355
6.152.2.36 MallocAndClearBlock() [6/11]	355
6.152.2.37 MallocAndClearBlock() [7/11]	355
6.152.2.38 MallocAndClearBlock() [8/11]	356
6.152.2.39 MallocAndClearBlock() [9/11]	356
6.152.2.40 MallocAndClearBlock() [10/11]	357
6.152.2.41 MallocAndClearBlock() [11/11]	357
6.152.2.42 MallocAndClearPtrArray< T >()	357
6.152.2.43 MallocAndClearPtrArrayMin1< T >()	358
6.152.2.44 MemClear()	358
6.152.2.45 MemCmp()	358
6.152.2.46 MemCpy()	359
6.152.2.47 MemCpyFast()	359
6.152.2.48 MemMove()	360
6.152.2.49 ReadLengthPrefixedUTF8()	360
6.152.2.50 ReferenceToPointer< T >()	360
6.152.2.51 RoundBitsUpTo32()	361
6.152.2.52 RoundBitsUpTo64()	361
6.152.2.53 RoundToAlignment() [1/2]	362
6.152.2.54 RoundToAlignment() [2/2]	362
6.152.2.55 RoundToMaxAlignment()	363
6.152.2.56 SizeOf()	363
6.152.2.57 WordCount()	363
6.152.2.58 WriteLengthPrefixedUTF8()	364
6.152.3 Member Data Documentation	364
6.152.3.1 ALIGNMENT	364
6.152.3.2 CACHE_LINE_SIZE	364
6.153 NestedComponentUtilities Class Reference	364
6.153.1 Detailed Description	365
6.153.2 Member Function Documentation	366
6.153.2.1 EnsureRootComponentExists< T, TStopOn >()	366
6.153.2.2 FindObjectsOfTypeInOrder< T >() [1/2]	366
6.153.2.3 FindObjectsOfTypeInOrder< T >() [2/2]	367
6.153.2.4 FindObjectsOfTypeInOrder< T, TCast >() [1/2]	368

6.153.2.5 FindObjectsOfTypeInOrder< T, TCast >() [2/2]	368
6.153.2.6 GetNestedComponentInChildren< T, TStopOn >()	369
6.153.2.7 GetNestedComponentInParent< T, TStopOn >()	370
6.153.2.8 GetNestedComponentInParents< T, TStopOn >()	370
6.153.2.9 GetNestedComponentsInChildren< T >()	371
6.153.2.10 GetNestedComponentsInChildren< T, TSearch, TStop >()	371
6.153.2.11 GetNestedComponentsInChildren< T, TStopOn >()	371
6.153.2.12 GetNestedComponentsInParents< T >()	372
6.153.2.13 GetNestedComponentsInParents< T, TStop >()	372
6.153.2.14 GetParentComponent< T >()	372
6.154 NetworkArray< T > Struct Template Reference	372
6.154.1 Detailed Description	374
6.154.2 Constructor & Destructor Documentation	374
6.154.2.1 NetworkArray()	374
6.154.3 Member Function Documentation	374
6.154.3.1 Clear()	375
6.154.3.2 CopyFrom() [1/2]	375
6.154.3.3 CopyFrom() [2/2]	375
6.154.3.4 CopyTo() [1/3]	376
6.154.3.5 CopyTo() [2/3]	376
6.154.3.6 CopyTo() [3/3]	376
6.154.3.7 Get()	377
6.154.3.8 GetEnumerator()	377
6.154.3.9 operator NetworkArrayReadOnly< T >()	377
6.154.3.10 Set()	377
6.154.3.11 ToArray()	378
6.154.3.12 ToListString()	378
6.154.3.13 ToReadOnly()	378
6.154.3.14 ToString()	378
6.154.4 Property Documentation	378
6.154.4.1 Length	378
6.154.4.2 this[int index]	378
6.155 NetworkArray< T >.Enumerator Struct Reference	379
6.155.1 Detailed Description	379
6.155.2 Constructor & Destructor Documentation	379
6.155.2.1 Enumerator()	379
6.155.3 Member Function Documentation	380
6.155.3.1 Dispose()	380
6.155.3.2 MoveNext()	380
6.155.3.3 Reset()	380
6.155.4 Property Documentation	380
6.155.4.1 Current [1/2]	380

6.155.4.2 Current [2/2]	381
6.156 NetworkArrayExtensions Class Reference	381
6.156.1 Detailed Description	381
6.156.2 Member Function Documentation	381
6.156.2.1 GetRef< T >()	381
6.156.2.2 IndexOf< T >()	382
6.157 NetworkArrayReadOnly< T > Struct Template Reference	382
6.157.1 Detailed Description	382
6.157.2 Property Documentation	383
6.157.2.1 Length	383
6.157.2.2 this[int index]	383
6.158 NetworkAssemblyIgnoreAttribute Class Reference	383
6.158.1 Detailed Description	383
6.159 NetworkAssemblyWeavedAttribute Class Reference	383
6.159.1 Detailed Description	383
6.160 NetworkBehaviour Class Reference	383
6.160.1 Detailed Description	387
6.160.2 Member Function Documentation	387
6.160.2.1 CopyBackingFieldsToState()	387
6.160.2.2 CopyStateFrom()	387
6.160.2.3 CopyStateToBackingFields()	387
6.160.2.4 Despawned()	388
6.160.2.5 FixedUpdateNetwork()	388
6.160.2.6 GetArrayReader< T >() [1/2]	388
6.160.2.7 GetArrayReader< T >() [2/2]	389
6.160.2.8 GetBehaviourReader< T >() [1/2]	389
6.160.2.9 GetBehaviourReader< T >() [2/2]	390
6.160.2.10 GetBehaviourReader< TBehaviour, TProperty >()	390
6.160.2.11 GetChangeDetector()	391
6.160.2.12 GetDictionaryReader< K, V >() [1/2]	391
6.160.2.13 GetDictionaryReader< K, V >() [2/2]	392
6.160.2.14 GetInput< T >() [1/2]	392
6.160.2.15 GetInput< T >() [2/2]	393
6.160.2.16 GetLinkListReader< T >() [1/2]	393
6.160.2.17 GetLinkListReader< T >() [2/2]	393
6.160.2.18 GetLocalAuthorityMask()	394
6.160.2.19 GetPropertyReader< T >() [1/2]	394
6.160.2.20 GetPropertyReader< T >() [2/2]	395
6.160.2.21 GetPropertyReader< TBehaviour, TProperty >()	395
6.160.2.22 MakeInitializer< K, V >()	396
6.160.2.23 MakeInitializer< T >()	396
6.160.2.24 MakePtr< T >() [1/2]	396

6.160.2.25 MakePtr< T >() [2/2]	397
6.160.2.26 MakeRef< T >() [1/2]	397
6.160.2.27 MakeRef< T >() [2/2]	398
6.160.2.28 NetworkDeserialize()	398
6.160.2.29 NetworkSerialize()	399
6.160.2.30 NetworkUnwrap()	399
6.160.2.31 NetworkWrap() [1/2]	400
6.160.2.32 NetworkWrap() [2/2]	400
6.160.2.33 operator NetworkBehaviourId()	400
6.160.2.34 ReinterpretState< T >()	401
6.160.2.35 ReplicateTo() [1/2]	401
6.160.2.36 ReplicateTo() [2/2]	402
6.160.2.37 ReplicateToAll()	402
6.160.2.38 ResetState()	402
6.160.2.39 Spawned()	403
6.160.2.40 TryGetSnapshotsBuffers()	403
6.160.3 Member Data Documentation	403
6.160.3.1 offset	403
6.160.4 Property Documentation	403
6.160.4.1 ChangedTick	404
6.160.4.2 DynamicWordCount	404
6.160.4.3 HasInputAuthority	404
6.160.4.4 HasStateAuthority	404
6.160.4.5 Id	404
6.160.4.6 IsProxy	404
6.160.4.7 StateBuffer	405
6.160.4.8 StateBufferIsValid	405
6.161 NetworkBehaviour.ArrayReader< T > Struct Template Reference	405
6.161.1 Detailed Description	405
6.161.2 Member Function Documentation	405
6.161.2.1 Read()	405
6.162 NetworkBehaviour.BehaviourReader< T > Struct Template Reference	406
6.162.1 Detailed Description	406
6.162.2 Member Function Documentation	406
6.162.2.1 Read()	406
6.162.3 Member Data Documentation	407
6.162.3.1 T	407
6.163 NetworkBehaviour.ChangeDetector Class Reference	407
6.163.1 Detailed Description	408
6.163.2 Member Enumeration Documentation	408
6.163.2.1 Source	408
6.163.3 Member Function Documentation	408

6.163.3.1 DetectChanges() [1/2]	408
6.163.3.2 DetectChanges() [2/2]	409
6.163.3.3 Init()	409
6.164 NetworkBehaviour.ChangeDetector.Enumerable Struct Reference	410
6.164.1 Detailed Description	410
6.164.2 Member Function Documentation	410
6.164.2.1 Changed()	410
6.164.2.2 GetEnumerator()	410
6.165 NetworkBehaviour.ChangeDetector.Enumerator Struct Reference	411
6.165.1 Detailed Description	411
6.165.2 Member Function Documentation	411
6.165.2.1 MoveNext()	411
6.165.2.2 Reset()	411
6.165.3 Property Documentation	412
6.165.3.1 Current	412
6.166 NetworkBehaviour.DictionaryReader< K, V > Struct Template Reference	412
6.166.1 Detailed Description	412
6.166.2 Member Function Documentation	412
6.166.2.1 Read()	412
6.167 NetworkBehaviour.LinkListReader< T > Struct Template Reference	413
6.167.1 Detailed Description	413
6.167.2 Member Function Documentation	413
6.167.2.1 Read()	413
6.168 NetworkBehaviour.PropertyReader< T > Struct Template Reference	414
6.168.1 Detailed Description	414
6.168.2 Constructor & Destructor Documentation	414
6.168.2.1 PropertyReader()	414
6.168.3 Member Function Documentation	415
6.168.3.1 Read()	415
6.168.4 Member Data Documentation	415
6.168.4.1 T	415
6.169 NetworkBehaviour.Buffer Struct Reference	415
6.169.1 Detailed Description	416
6.169.2 Member Function Documentation	416
6.169.2.1 operator bool()	416
6.169.2.2 Read() [1/5]	417
6.169.2.3 Read() [2/5]	417
6.169.2.4 Read() [3/5]	417
6.169.2.5 Read() [4/5]	419
6.169.2.6 Read() [5/5]	419
6.169.2.7 Read< T >() [1/2]	419
6.169.2.8 Read< T >() [2/2]	421

6.169.2.9 ReinterpretState< T >()	421
6.169.3 Property Documentation	422
6.169.3.1 Length	422
6.169.3.2 this[int index]	422
6.169.3.3 Tick	422
6.169.3.4 Valid	423
6.170 NetworkBehaviourBufferInterpolator Struct Reference	423
6.170.1 Detailed Description	424
6.170.2 Constructor & Destructor Documentation	424
6.170.2.1 NetworkBehaviourBufferInterpolator()	424
6.170.3 Member Function Documentation	425
6.170.3.1 Angle() [1/2]	425
6.170.3.2 Angle() [2/2]	425
6.170.3.3 Bool() [1/2]	425
6.170.3.4 Bool() [2/2]	426
6.170.3.5 Float() [1/2]	426
6.170.3.6 Float() [2/2]	426
6.170.3.7 Int() [1/2]	428
6.170.3.8 Int() [2/2]	428
6.170.3.9 operator bool()	428
6.170.3.10 Quaternion() [1/2]	429
6.170.3.11 Quaternion() [2/2]	429
6.170.3.12 Select< T >() [1/2]	430
6.170.3.13 Select< T >() [2/2]	430
6.170.3.14 Vector2() [1/2]	431
6.170.3.15 Vector2() [2/2]	431
6.170.3.16 Vector3() [1/2]	431
6.170.3.17 Vector3() [2/2]	432
6.170.3.18 Vector4() [1/2]	432
6.170.3.19 Vector4() [2/2]	432
6.170.4 Member Data Documentation	433
6.170.4.1 Alpha	433
6.170.4.2 Behaviour	433
6.170.4.3 From	433
6.170.4.4 To	433
6.170.4.5 Valid	433
6.171 NetworkBehaviourId Struct Reference	434
6.171.1 Detailed Description	435
6.171.2 Member Function Documentation	435
6.171.2.1 Equals() [1/2]	435
6.171.2.2 Equals() [2/2]	435
6.171.2.3 GetHashCode()	435

6.171.2.4 operator"!=()	436
6.171.2.5 operator==(())	436
6.171.2.6 ToString()	437
6.171.3 Member Data Documentation	437
6.171.3.1 Behaviour	437
6.171.3.2 Object	437
6.171.3.3 SIZE	437
6.171.4 Property Documentation	437
6.171.4.1 IsValid	437
6.171.4.2 None	438
6.172 NetworkBehaviourUtils Class Reference	438
6.172.1 Detailed Description	439
6.172.2 Member Function Documentation	440
6.172.2.1 CloneArray< T >()	440
6.172.2.2 CopyFromNetworkArray< T >()	440
6.172.2.3 CopyFromNetworkDictionary< D, K, V >()	441
6.172.2.4 CopyFromNetworkList< T >()	441
6.172.2.5 GetMetaData()	442
6.172.2.6 GetRpcStaticIndexOrThrow()	442
6.172.2.7 GetStaticWordCount()	443
6.172.2.8 GetWordCount()	443
6.172.2.9 HasStaticWordCount()	444
6.172.2.10 InitializeNetworkArray< T >()	444
6.172.2.11 InitializeNetworkDictionary< D, K, V >()	445
6.172.2.12 InitializeNetworkList< T >()	445
6.172.2.13 InternalOnDestroy()	446
6.172.2.14 InternalOnDisable()	446
6.172.2.15 InternalOnEnable()	446
6.172.2.16 MakeSerializableDictionary< K, V >()	447
6.172.2.17 NotifyLocalSimulationNotAllowedToSendRpc()	447
6.172.2.18 NotifyLocalTargetedRpcCulled()	448
6.172.2.19 NotifyNetworkUnwrapFailed< T >()	448
6.172.2.20 NotifyNetworkWrapFailed< T >() [1/2]	448
6.172.2.21 NotifyNetworkWrapFailed< T >() [2/2]	449
6.172.2.22 NotifyRpcPayloadSizeExceeded()	449
6.172.2.23 NotifyRpcTargetUnreachable()	449
6.172.2.24 RegisterMetaData()	450
6.172.2.25 RegisterRpcInvokeDelegates()	450
6.172.2.26 ShouldRegisterRpcInvokeDelegates()	450
6.172.2.27 ThrowIfBehaviourNotInitialized()	451
6.172.2.28 TryGetRpcInvokeDelegateArray()	451
6.172.2.29 TryGetRpcStaticInvokeDelegate()	452

6.172.3 Member Data Documentation	452
6.172.3.1 InvokeRpc	452
6.173 NetworkBehaviourUtils.ArrayInitializer< T > Struct Template Reference	452
6.173.1 Detailed Description	452
6.173.2 Member Function Documentation	453
6.173.2.1 operator NetworkArray< T >()	453
6.173.2.2 operator NetworkLinkedList< T >()	453
6.174 NetworkBehaviourUtils.DictionaryInitializer< K, V > Struct Template Reference	454
6.174.1 Detailed Description	454
6.174.2 Member Function Documentation	454
6.174.2.1 operator NetworkDictionary< K, V >()	454
6.175 NetworkBehaviourUtils.MetaData Struct Reference	455
6.175.1 Detailed Description	455
6.176 NetworkBehaviourWeavedAttribute Class Reference	455
6.176.1 Detailed Description	455
6.176.2 Constructor & Destructor Documentation	455
6.176.2.1 NetworkBehaviourWeavedAttribute()	455
6.176.3 Property Documentation	456
6.176.3.1 WordCount	456
6.177 NetworkBool Struct Reference	456
6.177.1 Detailed Description	457
6.177.2 Constructor & Destructor Documentation	457
6.177.2.1 NetworkBool()	457
6.177.3 Member Function Documentation	457
6.177.3.1 Equals() [1/2]	457
6.177.3.2 Equals() [2/2]	458
6.177.3.3 GetHashCode()	458
6.177.3.4 operator bool()	458
6.177.3.5 operator NetworkBool()	458
6.177.3.6 ToString()	459
6.177.4 Member Data Documentation	459
6.177.4.1 SIZE	459
6.178 NetworkButtons Struct Reference	459
6.178.1 Detailed Description	460
6.178.2 Constructor & Destructor Documentation	460
6.178.2.1 NetworkButtons()	461
6.178.3 Member Function Documentation	461
6.178.3.1 Equals() [1/2]	461
6.178.3.2 Equals() [2/2]	461
6.178.3.3 GetHashCode()	462
6.178.3.4 GetPressed()	462
6.178.3.5 GetReleased()	462

6.178.3.6 IsSet()	463
6.178.3.7 IsSet< T >()	463
6.178.3.8 Set()	463
6.178.3.9 Set< T >()	464
6.178.3.10 SetAllDown()	464
6.178.3.11 SetAllUp()	464
6.178.3.12 SetDown()	464
6.178.3.13 SetDown< T >()	465
6.178.3.14 SetUp()	465
6.178.3.15 SetUp< T >()	465
6.178.3.16 WasPressed()	466
6.178.3.17 WasPressed< T >()	466
6.178.3.18 WasReleased()	467
6.178.3.19 WasReleased< T >()	467
6.178.4 Member Data Documentation	468
6.178.4.1 NetworkButtons	468
6.178.5 Property Documentation	468
6.178.5.1 Bits	468
6.179 NetworkConfiguration Class Reference	468
6.179.1 Detailed Description	469
6.179.2 Member Enumeration Documentation	469
6.179.2.1 ReliableDataTransfers	469
6.179.3 Member Function Documentation	470
6.179.3.1 Init()	470
6.179.4 Member Data Documentation	470
6.179.4.1 ConnectionShutdownTime	470
6.179.4.2 ConnectionTimeout	470
6.179.4.3 ReliableDataTransferModes	471
6.179.5 Property Documentation	471
6.179.5.1 ConnectAttempts	471
6.179.5.2 ConnectInterval	471
6.179.5.3 ConnectionDefaultRtt	471
6.179.5.4 ConnectionPingInterval	471
6.179.5.5 SocketRecvBufferSize	472
6.179.5.6 SocketSendBufferSize	472
6.180 NetworkDelegates Class Reference	472
6.180.1 Detailed Description	472
6.181 NetworkDeserializeMethodAttribute Class Reference	473
6.181.1 Detailed Description	473
6.182 NetworkDictionary< K, V > Struct Template Reference	473
6.182.1 Detailed Description	475
6.182.2 Constructor & Destructor Documentation	475

6.182.2.1 NetworkDictionary()	475
6.182.3 Member Function Documentation	476
6.182.3.1 Add() [1/2]	476
6.182.3.2 Add() [2/2]	476
6.182.3.3 Clear()	476
6.182.3.4 ContainsKey()	476
6.182.3.5 ContainsValue()	476
6.182.3.6 Get()	477
6.182.3.7 GetEnumerator()	477
6.182.3.8 operator NetworkDictionaryReadOnly< K, V >()	477
6.182.3.9 Remove() [1/2]	477
6.182.3.10 Remove() [2/2]	478
6.182.3.11 Set()	478
6.182.3.12 ToReadOnly()	478
6.182.3.13 TryGet()	479
6.182.4 Member Data Documentation	479
6.182.4.1 META_WORD_COUNT	479
6.182.5 Property Documentation	479
6.182.5.1 Capacity	479
6.182.5.2 Count	479
6.182.5.3 this[K key]	480
6.183 NetworkDictionary< K, V >.Enumerator Struct Reference	480
6.183.1 Detailed Description	480
6.183.2 Member Function Documentation	480
6.183.2.1 Dispose()	481
6.183.2.2 MoveNext()	481
6.183.2.3 Reset()	481
6.183.3 Property Documentation	481
6.183.3.1 Current	481
6.184 NetworkDictionaryReadOnly< K, V > Struct Template Reference	481
6.184.1 Detailed Description	482
6.184.2 Member Function Documentation	483
6.184.2.1 Get()	483
6.184.2.2 TryGet()	483
6.184.3 Property Documentation	483
6.184.3.1 Capacity	483
6.184.3.2 Count	484
6.185 NetworkedAttribute Class Reference	484
6.185.1 Detailed Description	484
6.185.2 Constructor & Destructor Documentation	484
6.185.2.1 NetworkedAttribute()	484
6.185.3 Property Documentation	484

6.185.3.1 Default	485
6.186 NetworkedWeavedArrayAttribute Class Reference	485
6.186.1 Detailed Description	485
6.186.2 Property Documentation	485
6.186.2.1 Capacity	485
6.186.2.2 ElementReaderWriterType	486
6.186.2.3 ElementWordCount	486
6.187 NetworkedWeavedAttribute Class Reference	486
6.187.1 Detailed Description	486
6.187.2 Constructor & Destructor Documentation	486
6.187.2.1 NetworkedWeavedAttribute()	486
6.187.3 Property Documentation	487
6.187.3.1 WordCount	487
6.187.3.2 WordOffset	487
6.188 NetworkedWeavedDictionaryAttribute Class Reference	487
6.188.1 Detailed Description	488
6.188.2 Property Documentation	488
6.188.2.1 Capacity	488
6.188.2.2 KeyWordCount	488
6.188.2.3 ValueWordCount	488
6.189 NetworkedWeavedLinkedListAttribute Class Reference	488
6.189.1 Detailed Description	489
6.189.2 Property Documentation	489
6.189.2.1 Capacity	489
6.189.2.2 ElementReaderWriterType	489
6.189.2.3 ElementWordCount	489
6.190 NetworkEvents Class Reference	489
6.190.1 Detailed Description	491
6.191 NetworkEvents.ConnectFailedEvent Class Reference	491
6.191.1 Detailed Description	491
6.192 NetworkEvents.ConnectRequestEvent Class Reference	491
6.192.1 Detailed Description	491
6.193 NetworkEvents.CustomAuthenticationResponse Class Reference	491
6.193.1 Detailed Description	491
6.194 NetworkEvents.DisconnectFromServerEvent Class Reference	491
6.194.1 Detailed Description	492
6.195 NetworkEvents.HostMigrationEvent Class Reference	492
6.195.1 Detailed Description	492
6.196 NetworkEvents.InputEvent Class Reference	492
6.196.1 Detailed Description	492
6.197 NetworkEvents.InputPlayerEvent Class Reference	492
6.197.1 Detailed Description	492

6.198 NetworkEvents.ObjectEvent Class Reference	492
6.198.1 Detailed Description	493
6.199 NetworkEvents.ObjectPlayerEvent Class Reference	493
6.199.1 Detailed Description	493
6.200 NetworkEvents.PlayerEvent Class Reference	493
6.200.1 Detailed Description	493
6.201 NetworkEvents.ReliableDataEvent Class Reference	493
6.201.1 Detailed Description	493
6.202 NetworkEvents.ReliableProgressEvent Class Reference	493
6.202.1 Detailed Description	494
6.203 NetworkEvents.RunnerEvent Class Reference	494
6.203.1 Detailed Description	494
6.204 NetworkEvents.SessionListUpdateEvent Class Reference	494
6.204.1 Detailed Description	494
6.205 NetworkEvents.ShutdownEvent Class Reference	494
6.205.1 Detailed Description	494
6.206 NetworkEvents.SimulationMessageEvent Class Reference	494
6.206.1 Detailed Description	495
6.207 NetworkId Struct Reference	495
6.207.1 Detailed Description	496
6.207.2 Member Function Documentation	496
6.207.2.1 CompareTo()	496
6.207.2.2 Equals() [1/2]	497
6.207.2.3 Equals() [2/2]	497
6.207.2.4 GetHashCode()	497
6.207.2.5 operator bool()	498
6.207.2.6 operator"!=()	498
6.207.2.7 operator==(())	498
6.207.2.8 Read()	498
6.207.2.9 ToNamePrefixString()	498
6.207.2.10 ToString()	499
6.207.2.11 Write() [1/2]	499
6.207.2.12 Write() [2/2]	499
6.207.3 Member Data Documentation	499
6.207.3.1 ALIGNMENT	499
6.207.3.2 BLOCK_SIZE	500
6.207.3.3 Raw	500
6.207.3.4 SIZE	500
6.207.4 Property Documentation	500
6.207.4.1 Comparer	500
6.207.4.2 IsReserved	500
6.207.4.3 IsValid	500

6.208 NetworkId.EqualityComparer Class Reference	501
6.208.1 Detailed Description	501
6.208.2 Member Function Documentation	501
6.208.2.1 Equals()	501
6.208.2.2 GetHashCode()	501
6.209 NetworkInput Struct Reference	501
6.209.1 Detailed Description	502
6.209.2 Member Function Documentation	502
6.209.2.1 Convert< T >()	502
6.209.2.2 Get< T >()	503
6.209.2.3 Is< T >()	503
6.209.2.4 Set< T >()	503
6.209.2.5 TryGet< T >()	503
6.209.2.6 TrySet< T >()	504
6.209.3 Property Documentation	504
6.209.3.1 Data	504
6.209.3.2 IsValid	504
6.209.3.3 Type	504
6.209.3.4 WordCount	504
6.210 NetworkInputUtils Class Reference	505
6.210.1 Detailed Description	505
6.210.2 Member Function Documentation	505
6.210.2.1 GetMaxWordCount()	505
6.210.2.2 GetType()	505
6.210.2.3 GetTypeKey()	506
6.210.2.4 GetWordCount()	506
6.211 NetworkInputWeavedAttribute Class Reference	506
6.211.1 Detailed Description	507
6.211.2 Constructor & Destructor Documentation	507
6.211.2.1 NetworkInputWeavedAttribute()	507
6.211.3 Property Documentation	507
6.211.3.1 WordCount	507
6.212 NetworkLinkedList< T > Struct Template Reference	507
6.212.1 Detailed Description	509
6.212.2 Constructor & Destructor Documentation	509
6.212.2.1 NetworkLinkedList()	510
6.212.3 Member Function Documentation	510
6.212.3.1 Add() [1/2]	510
6.212.3.2 Add() [2/2]	510
6.212.3.3 Clear()	511
6.212.3.4 Contains() [1/2]	511
6.212.3.5 Contains() [2/2]	511

6.212.3.6 Get()	511
6.212.3.7 GetEnumerator()	511
6.212.3.8 IndexOf() [1/2]	511
6.212.3.9 IndexOf() [2/2]	511
6.212.3.10 Remap()	512
6.212.3.11 Remove() [1/2]	512
6.212.3.12 Remove() [2/2]	512
6.212.3.13 Set()	513
6.212.4 Member Data Documentation	513
6.212.4.1 ELEMENT_WORDS	513
6.212.4.2 META_WORDS	513
6.212.5 Property Documentation	513
6.212.5.1 Capacity	513
6.212.5.2 Count	513
6.212.5.3 this[int index]	514
6.213 NetworkLinkedList< T >.Enumerator Struct Reference	514
6.213.1 Detailed Description	514
6.213.2 Member Function Documentation	514
6.213.2.1 Dispose()	515
6.213.2.2 MoveNext()	515
6.213.2.3 Reset()	515
6.213.3 Property Documentation	515
6.213.3.1 Current	515
6.214 NetworkLinkedListReadOnly< T > Struct Template Reference	516
6.214.1 Detailed Description	517
6.214.2 Member Function Documentation	518
6.214.2.1 Contains() [1/2]	518
6.214.2.2 Contains() [2/2]	518
6.214.2.3 Get()	518
6.214.2.4 IndexOf() [1/2]	518
6.214.2.5 IndexOf() [2/2]	518
6.214.3 Member Data Documentation	519
6.214.3.1 ELEMENT_WORDS	519
6.214.3.2 META_WORDS	519
6.214.4 Property Documentation	519
6.214.4.1 Capacity	519
6.214.4.2 Count	519
6.214.4.3 this[int index]	520
6.215 NetworkLoadSceneParameters Struct Reference	520
6.215.1 Detailed Description	521
6.215.2 Member Function Documentation	521
6.215.2.1 Equals() [1/2]	521

6.215.2.2 Equals() [2/2]	521
6.215.2.3 GetHashCode()	521
6.215.2.4 operator"!="()	522
6.215.2.5 operator=="()	522
6.215.2.6 ToString()	522
6.215.3 Member Data Documentation	522
6.215.3.1 LoadId	523
6.215.4 Property Documentation	523
6.215.4.1 IsActiveOnLoad	523
6.215.4.2 IsLocalPhysics2D	523
6.215.4.3 IsLocalPhysics3D	523
6.215.4.4 IsSingleLoad	523
6.215.4.5 LoadSceneMode	523
6.215.4.6 LoadSceneParameters	524
6.215.4.7 LocalPhysicsMode	524
6.216 NetworkMecanimAnimator Class Reference	524
6.216.1 Detailed Description	525
6.216.2 Member Function Documentation	525
6.216.2.1 SetTrigger() [1/2]	525
6.216.2.2 SetTrigger() [2/2]	525
6.216.3 Member Data Documentation	526
6.216.3.1 Animator	526
6.216.3.2 ApplyTiming	526
6.216.4 Property Documentation	526
6.216.4.1 DynamicWordCount	526
6.217 NetworkObject Class Reference	527
6.217.1 Detailed Description	529
6.217.2 Member Function Documentation	529
6.217.2.1 AssignInputAuthority()	529
6.217.2.2 Awake()	529
6.217.2.3 CopyStateFrom() [1/2]	529
6.217.2.4 CopyStateFrom() [2/2]	530
6.217.2.5 GetLocalAuthorityMask()	530
6.217.2.6 GetWordCount()	530
6.217.2.7 NetworkUnwrap()	531
6.217.2.8 NetworkWrap() [1/2]	531
6.217.2.9 NetworkWrap() [2/2]	531
6.217.2.10 OnDestroy()	532
6.217.2.11 operator NetworkId()	532
6.217.2.12 PriorityLevelDelegate()	532
6.217.2.13 ReleaseStateAuthority()	533
6.217.2.14 RemoveInputAuthority()	533

6.217.2.15 ReplicateToDelegate()	533
6.217.2.16 RequestStateAuthority()	533
6.217.2.17 SetPlayerAlwaysInterested()	533
6.217.3 Member Data Documentation	534
6.217.3.1 Flags	534
6.217.3.2 ForceRemoteRenderTimeframe	534
6.217.3.3 IsResume	534
6.217.3.4 NestedObjects	534
6.217.3.5 NetworkedBehaviours	534
6.217.3.6 NetworkTypeId	535
6.217.3.7 PriorityCallback	535
6.217.3.8 ReplicateTo	535
6.217.3.9 SortKey	535
6.217.4 Property Documentation	535
6.217.4.1 HasInputAuthority	535
6.217.4.2 HasStateAuthority	536
6.217.4.3 Id	536
6.217.4.4 InputAuthority	536
6.217.4.5 IsInSimulation	536
6.217.4.6 IsNested	536
6.217.4.7 IsProxy	536
6.217.4.8 IsSpawnable	537
6.217.4.9 IsValid	537
6.217.4.10 LastReceiveTick	537
6.217.4.11 Name	537
6.217.4.12 NestingRoot	537
6.217.4.13 RenderSource	537
6.217.4.14 RenderTime	538
6.217.4.15 RenderTimeframe	538
6.217.4.16 Runner	538
6.217.4.17 StateAuthority	538
6.218 NetworkObjectFlagsExtensions Class Reference	538
6.218.1 Detailed Description	539
6.218.2 Member Function Documentation	539
6.218.2.1 GetVersion()	539
6.218.2.2 IsIgnored()	539
6.218.2.3 IsVersionCurrent()	539
6.218.2.4 SetCurrentVersion()	540
6.218.2.5 SetIgnored()	540
6.219 NetworkObjectGuid Struct Reference	540
6.219.1 Detailed Description	542
6.219.2 Constructor & Destructor Documentation	542

6.219.2.1 NetworkObjectGuid() [1/4]	542
6.219.2.2 NetworkObjectGuid() [2/4]	542
6.219.2.3 NetworkObjectGuid() [3/4]	544
6.219.2.4 NetworkObjectGuid() [4/4]	544
6.219.3 Member Function Documentation	544
6.219.3.1 CompareTo()	544
6.219.3.2 Equals() [1/2]	545
6.219.3.3 Equals() [2/2]	545
6.219.3.4 GetHashCode()	545
6.219.3.5 operator Guid()	546
6.219.3.6 operator NetworkObjectGuid()	546
6.219.3.7 operator NetworkPrefabRef()	546
6.219.3.8 operator "!="()	547
6.219.3.9 operator "=="()	547
6.219.3.10 Parse()	547
6.219.3.11 ToString() [1/2]	548
6.219.3.12 ToString() [2/2]	548
6.219.3.13 ToUnityGuidString()	548
6.219.3.14 TryParse()	548
6.219.4 Member Data Documentation	548
6.219.4.1 ALIGNMENT	549
6.219.4.2 RawGuidValue	549
6.219.4.3 SIZE	549
6.219.5 Property Documentation	549
6.219.5.1 Empty	549
6.219.5.2 IsValid	549
6.220 NetworkObjectGuid.EqualityComparer Class Reference	549
6.220.1 Detailed Description	550
6.220.2 Member Function Documentation	550
6.220.2.1 Equals()	550
6.220.2.2 GetHashCode()	550
6.221 NetworkObjectHeader Struct Reference	550
6.221.1 Detailed Description	552
6.221.2 Member Function Documentation	552
6.221.2.1 Equals() [1/2]	552
6.221.2.2 Equals() [2/2]	552
6.221.2.3 GetBehaviourChangedTickArray()	552
6.221.2.4 GetDataPointer()	553
6.221.2.5 GetDataWordCount()	553
6.221.2.6 GetHashCode()	553
6.221.2.7 GetMainNetworkTRSPData()	553
6.221.2.8 HasMainNetworkTRSP()	554

6.221.2.9 operator"!=()	554
6.221.2.10 operator==()	554
6.221.2.11 ToString()	555
6.221.3 Member Data Documentation	555
6.221.3.1 _reserved	555
6.221.3.2 BehaviourCount	555
6.221.3.3 Flags	555
6.221.3.4 Id	555
6.221.3.5 InputAuthority	555
6.221.3.6 NestingKey	556
6.221.3.7 NestingRoot	556
6.221.3.8 PLAYER_DATA_WORD	556
6.221.3.9 SIZE	556
6.221.3.10 StateAuthority	556
6.221.3.11 Type	556
6.221.3.12 WordCount	557
6.221.3.13 WORDS	557
6.221.4 Property Documentation	557
6.221.4.1 ByteCount	557
6.222 NetworkObjectHeaderPtr Struct Reference	557
6.222.1 Detailed Description	558
6.222.2 Member Function Documentation	558
6.222.2.1 operator NetworkObjectHeaderPtr()	558
6.222.3 Member Data Documentation	558
6.222.3.1 Ptr	558
6.222.4 Property Documentation	558
6.222.4.1 Id	558
6.222.4.2 Type	559
6.223 NetworkObjectInitializerUnity Class Reference	559
6.223.1 Detailed Description	559
6.223.2 Member Function Documentation	559
6.223.2.1 InitializeNetworkState()	559
6.224 NetworkObjectMeta Class Reference	559
6.224.1 Detailed Description	560
6.224.2 Property Documentation	560
6.224.2.1 Flags	560
6.224.2.2 Id	560
6.224.2.3 InputAuthority	560
6.224.2.4 NestingKey	561
6.224.2.5 NestingRoot	561
6.224.2.6 StateAuthority	561
6.224.2.7 Type	561

6.225 NetworkObjectNestingKey Struct Reference	561
6.225.1 Detailed Description	562
6.225.2 Constructor & Destructor Documentation	562
6.225.2.1 NetworkObjectNestingKey()	562
6.225.3 Member Function Documentation	563
6.225.3.1 Equals() [1/2]	563
6.225.3.2 Equals() [2/2]	563
6.225.3.3 GetHashCode()	563
6.225.3.4 ToString()	564
6.225.4 Member Data Documentation	564
6.225.4.1 ALIGNMENT	564
6.225.4.2 SIZE	564
6.225.4.3 Value	564
6.225.5 Property Documentation	564
6.225.5.1 IsNone	565
6.225.5.2 IsValid	565
6.226 NetworkObjectNestingKey.EqualityComparer Class Reference	565
6.226.1 Detailed Description	565
6.226.2 Member Function Documentation	565
6.226.2.1 Equals()	566
6.226.2.2 GetHashCode()	566
6.227 NetworkObjectPrefabData Class Reference	566
6.227.1 Detailed Description	566
6.227.2 Member Data Documentation	566
6.227.2.1 Guid	566
6.228 NetworkObjectProviderDummy Class Reference	567
6.228.1 Detailed Description	567
6.229 NetworkObjectReleaseContext Struct Reference	567
6.229.1 Detailed Description	568
6.229.2 Constructor & Destructor Documentation	568
6.229.2.1 NetworkObjectReleaseContext()	568
6.229.3 Member Function Documentation	568
6.229.3.1 ToString()	568
6.229.4 Member Data Documentation	568
6.229.4.1 IsBeingDestroyed	569
6.229.4.2 IsNestedObject	569
6.229.4.3 Object	569
6.229.4.4 Typeld	569
6.230 NetworkObjectSortKeyComparer Class Reference	569
6.230.1 Detailed Description	570
6.230.2 Member Function Documentation	570
6.230.2.1 Compare()	570

6.230.3 Member Data Documentation	570
6.230.3.1 Instance	570
6.231 NetworkObjectSpawnException Class Reference	570
6.231.1 Detailed Description	571
6.231.2 Constructor & Destructor Documentation	571
6.231.2.1 NetworkObjectSpawnException()	571
6.231.3 Property Documentation	571
6.231.3.1 Message	571
6.231.3.2 Status	571
6.231.3.3 Typeld	572
6.232 NetworkObjectTypeld Struct Reference	572
6.232.1 Detailed Description	573
6.232.2 Member Function Documentation	574
6.232.2.1 Equals() [1/2]	574
6.232.2.2 Equals() [2/2]	574
6.232.2.3 FromCustom()	574
6.232.2.4 FromPrefabld()	575
6.232.2.5 FromSceneObjectld()	575
6.232.2.6 FromSceneRefAndObjectIndex()	575
6.232.2.7 FromStruct()	576
6.232.2.8 GetHashCode()	576
6.232.2.9 operator NetworkObjectTypeld()	576
6.232.2.10 operator"!=(577
6.232.2.11 operator==(577
6.232.2.12 ToString()	577
6.232.3 Member Data Documentation	577
6.232.3.1 _value0	577
6.232.3.2 _value1	578
6.232.3.3 ALIGNMENT	578
6.232.3.4 MAX_SCENE_OBJECT_INDEX	578
6.232.3.5 SIZE	578
6.232.4 Property Documentation	578
6.232.4.1 AsCustom	578
6.232.4.2 AsInternalStructld	579
6.232.4.3 AsPrefabld	579
6.232.4.4 AsSceneObjectld	579
6.232.4.5 Comparer	579
6.232.4.6 IsCustom	580
6.232.4.7 IsNone	580
6.232.4.8 IsPrefab	580
6.232.4.9 IsSceneObject	580
6.232.4.10 IsStruct	580

6.232.4.11 IsValid	580
6.232.4.12 Kind	581
6.232.4.13 PlayerData	581
6.233 NetworkObjectTypeId.EqualityComparer Class Reference	581
6.233.1 Detailed Description	581
6.233.2 Member Function Documentation	581
6.233.2.1 Equals()	581
6.233.2.2 GetHashCode()	581
6.234 NetworkPhysicsInfo Struct Reference	582
6.234.1 Detailed Description	582
6.234.2 Member Data Documentation	582
6.234.2.1 SIZE	582
6.234.2.2 TimeScale	582
6.234.2.3 WORD_COUNT	583
6.235 NetworkPrefabAcquireContext Struct Reference	583
6.235.1 Detailed Description	583
6.235.2 Constructor & Destructor Documentation	583
6.235.2.1 NetworkPrefabAcquireContext()	583
6.235.3 Member Data Documentation	584
6.235.3.1 DontDestroyOnLoad	584
6.235.3.2 IsSynchronous	584
6.235.3.3 Meta	584
6.235.3.4 PrefabId	584
6.235.4 Property Documentation	584
6.235.4.1 Data	584
6.235.4.2 HasHeader	585
6.236 NetworkPrefabAttribute Class Reference	585
6.236.1 Detailed Description	585
6.237 NetworkPrefabId Struct Reference	585
6.237.1 Detailed Description	587
6.237.2 Member Function Documentation	587
6.237.2.1 CompareTo() [1/2]	587
6.237.2.2 CompareTo() [2/2]	587
6.237.2.3 Equals() [1/2]	587
6.237.2.4 Equals() [2/2]	587
6.237.2.5 FromIndex()	588
6.237.2.6 FromRaw()	588
6.237.2.7 GetHashCode()	588
6.237.2.8 operator"!=()	588
6.237.2.9 operator==()	588
6.237.2.10 ToString() [1/2]	589
6.237.2.11 ToString() [2/2]	589

6.237.3 Member Data Documentation	589
6.237.3.1 ALIGNMENT	589
6.237.3.2 MAX_INDEX	589
6.237.3.3 RawValue	589
6.237.3.4 SIZE	589
6.237.4 Property Documentation	590
6.237.4.1 AsIndex	590
6.237.4.2 IsNone	590
6.237.4.3 IsValid	590
6.238 NetworkPrefabId.EqualityComparer Class Reference	590
6.238.1 Detailed Description	590
6.238.2 Member Function Documentation	591
6.238.2.1 Equals()	591
6.238.2.2 GetHashCode()	591
6.239 NetworkPrefabInfo Struct Reference	591
6.239.1 Detailed Description	591
6.239.2 Member Data Documentation	592
6.239.2.1 Header	592
6.239.2.2 IsSynchronous	592
6.239.2.3 Prefab	592
6.239.3 Property Documentation	592
6.239.3.1 Data	592
6.239.3.2 HasHeader	592
6.240 NetworkPrefabRef Struct Reference	593
6.240.1 Detailed Description	594
6.240.2 Constructor & Destructor Documentation	594
6.240.2.1 NetworkPrefabRef() [1/4]	594
6.240.2.2 NetworkPrefabRef() [2/4]	594
6.240.2.3 NetworkPrefabRef() [3/4]	595
6.240.2.4 NetworkPrefabRef() [4/4]	595
6.240.3 Member Function Documentation	595
6.240.3.1 CompareTo()	595
6.240.3.2 Equals() [1/2]	596
6.240.3.3 Equals() [2/2]	596
6.240.3.4 GetHashCode()	596
6.240.3.5 operator Guid()	597
6.240.3.6 operator NetworkObjectGuid()	597
6.240.3.7 operator NetworkPrefabRef()	597
6.240.3.8 operator "!="()	598
6.240.3.9 operator "=="()	598
6.240.3.10 Parse()	598
6.240.3.11 ToString() [1/2]	599

6.240.3.12 ToString() [2/2]	599
6.240.3.13 ToUnityGuidString()	599
6.240.3.14 TryParse()	599
6.240.4 Member Data Documentation	599
6.240.4.1 ALIGNMENT	600
6.240.4.2 RawGuidValue	600
6.240.4.3 SIZE	600
6.240.5 Property Documentation	600
6.240.5.1 Empty	600
6.240.5.2 IsValid	600
6.241 NetworkPrefabRef.EqualityComparer Class Reference	600
6.241.1 Detailed Description	601
6.241.2 Member Function Documentation	601
6.241.2.1 Equals()	601
6.241.2.2 GetHashCode()	601
6.242 NetworkPrefabTable Class Reference	601
6.242.1 Detailed Description	603
6.242.2 Member Function Documentation	603
6.242.2.1 AddInstance()	603
6.242.2.2 AddSource()	603
6.242.2.3 Clear()	603
6.242.2.4 Contains()	604
6.242.2.5 GetEntries()	604
6.242.2.6 GetGuid()	604
6.242.2.7 GetId()	605
6.242.2.8 GetInstancesCount()	605
6.242.2.9 GetSource() [1/2]	605
6.242.2.10 GetSource() [2/2]	606
6.242.2.11 IsAcquired()	606
6.242.2.12 Load()	606
6.242.2.13 RemoveInstance()	607
6.242.2.14 TryAddSource()	607
6.242.2.15 Unload()	608
6.242.2.16 UnloadAll()	608
6.242.2.17 UnloadUnreferenced()	608
6.242.3 Member Data Documentation	609
6.242.3.1 Options	609
6.242.4 Property Documentation	609
6.242.4.1 Prefabs	609
6.242.4.2 Version	609
6.243 NetworkPrefabTableOptions Struct Reference	609
6.243.1 Detailed Description	610

6.243.2 Member Data Documentation	610
6.243.2.1 Default	610
6.243.2.2 UnloadPrefabOnReleasingLastInstance	610
6.243.2.3 UnloadUnusedPrefabsOnShutdown	610
6.244 NetworkProjectConfig Class Reference	610
6.244.1 Detailed Description	612
6.244.2 Member Enumeration Documentation	613
6.244.2.1 PeerModes	613
6.244.2.2 ReplicationFeatures	613
6.244.3 Member Function Documentation	613
6.244.3.1 Deserialize()	613
6.244.3.2 GetExecutionOrder()	614
6.244.3.3 Serialize()	614
6.244.3.4 ToString()	615
6.244.3.5 UnloadGlobal()	615
6.244.4 Member Data Documentation	615
6.244.4.1 AllowClientServerModesInWebGL	615
6.244.4.2 AssembliesToWeave	615
6.244.4.3 BuildTypes	615
6.244.4.4 CheckNetworkedPropertiesBeingEmpty	616
6.244.4.5 CheckRpcAttributeUsage	616
6.244.4.6 CurrentTypeId	616
6.244.4.7 CurrentVersion	616
6.244.4.8 DefaultResourceName	616
6.244.4.9 EncryptionConfig	616
6.244.4.10 EnqueueIncompleteSynchronousSpawns	617
6.244.4.11 Heap	617
6.244.4.12 HideNetworkObjectInactivityGuard	617
6.244.4.13 HostMigration	617
6.244.4.14 InvokeRenderInBatchMode	617
6.244.4.15 LagCompensation	618
6.244.4.16 Network	618
6.244.4.17 NetworkConditions	618
6.244.4.18 NetworkIdsObjectName	618
6.244.4.19 NullChecksForNetworkedProperties	618
6.244.4.20 PeerMode	618
6.244.4.21 PrefabTable	619
6.244.4.22 Simulation	619
6.244.4.23 TimeSynchronizationOverride	619
6.244.4.24 TypeId	619
6.244.4.25 UseSerializableDictionary	619
6.244.4.26 Version	619

6.244.5 Property Documentation	620
6.244.5.1 Global	620
6.245 NetworkProjectConfigAsset Class Reference	620
6.245.1 Detailed Description	621
6.245.2 Member Function Documentation	621
6.245.2.1 OnDisable()	621
6.245.2.2 TryGetGlobal()	621
6.245.2.3 UnloadGlobal()	622
6.245.3 Member Data Documentation	622
6.245.3.1 BehaviourMeta	622
6.245.3.2 Config	622
6.245.3.3 PrefabOptions	622
6.245.3.4 Prefabs	622
6.245.4 Property Documentation	622
6.245.4.1 Global	623
6.245.4.2 IsGlobalLoaded	623
6.246 NetworkProjectConfigAsset.SerializableSimulationBehaviourMeta Struct Reference	623
6.246.1 Detailed Description	623
6.246.2 Member Data Documentation	623
6.246.2.1 ExecutionOrder	623
6.246.2.2 Type	624
6.247 NetworkRNG Struct Reference	624
6.247.1 Detailed Description	625
6.247.2 Constructor & Destructor Documentation	625
6.247.2.1 NetworkRNG()	625
6.247.3 Member Function Documentation	625
6.247.3.1 Next()	625
6.247.3.2 NextExclusive()	626
6.247.3.3 NextInt32()	626
6.247.3.4 NextSingle()	626
6.247.3.5 NextSingleExclusive()	626
6.247.3.6 NextUInt32()	627
6.247.3.7 RangeExclusive() [1/2]	627
6.247.3.8 RangeExclusive() [2/2]	627
6.247.3.9 RangeInclusive() [1/4]	627
6.247.3.10 RangeInclusive() [2/4]	628
6.247.3.11 RangeInclusive() [3/4]	628
6.247.3.12 RangeInclusive() [4/4]	628
6.247.3.13 ToString()	628
6.247.4 Member Data Documentation	628
6.247.4.1 MAX	628
6.247.4.2 SIZE	629

6.247.5 Property Documentation	629
6.247.5.1 Peek	629
6.248 NetworkRpcStaticWeavedInvokerAttribute Class Reference	629
6.248.1 Detailed Description	629
6.248.2 Constructor & Destructor Documentation	629
6.248.2.1 NetworkRpcStaticWeavedInvokerAttribute()	629
6.248.3 Property Documentation	630
6.248.3.1 Key	630
6.249 NetworkRpcWeavedInvokerAttribute Class Reference	630
6.249.1 Detailed Description	630
6.249.2 Constructor & Destructor Documentation	630
6.249.2.1 NetworkRpcWeavedInvokerAttribute()	630
6.249.3 Property Documentation	631
6.249.3.1 Key	631
6.249.3.2 Sources	631
6.249.3.3 Targets	631
6.250 NetworkRunner Class Reference	631
6.250.1 Detailed Description	641
6.250.2 Member Enumeration Documentation	641
6.250.2.1 BuildTypes	641
6.250.2.2 States	641
6.250.3 Member Function Documentation	641
6.250.3.1 AddCallbacks()	641
6.250.3.2 AddGlobal()	642
6.250.3.3 AddPlayerAreaOfInterest()	642
6.250.3.4 Attach() [1/2]	642
6.250.3.5 Attach() [2/2]	643
6.250.3.6 ClearPlayerAreaOfInterest()	643
6.250.3.7 CloudConnectionLostHandler()	643
6.250.3.8 Despawn()	644
6.250.3.9 DestroySingleton< T >()	644
6.250.3.10 Disconnect()	644
6.250.3.11 EnsureRunnerScenelsActive()	645
6.250.3.12 Exists() [1/2]	645
6.250.3.13 Exists() [2/2]	645
6.250.3.14 FindObject()	645
6.250.3.15 GetAllBehaviours()	646
6.250.3.16 GetAllBehaviours< T >() [1/2]	646
6.250.3.17 GetAllBehaviours< T >() [2/2]	646
6.250.3.18 GetAllNetworkObjects() [1/2]	647
6.250.3.19 GetAllNetworkObjects() [2/2]	647
6.250.3.20 GetAreaOfInterestGizmoData()	647

6.250.3.21 GetAvailableRegions()	648
6.250.3.22 GetInputForPlayer< T >()	648
6.250.3.23 GetInstancesEnumerator()	648
6.250.3.24 GetInterfaceListHead()	649
6.250.3.25 GetInterfaceListNext()	650
6.250.3.26 GetInterfaceListPrev()	650
6.250.3.27 GetInterfaceListsCount()	651
6.250.3.28 GetMemorySnapshot()	651
6.250.3.29 GetObjectsInAreaOfInterestForPlayer()	651
6.250.3.30 GetPhysicsScene()	652
6.250.3.31 GetPhysicsScene2D()	652
6.250.3.32 GetPlayerActorId()	652
6.250.3.33 GetPlayerConnectionToken()	652
6.250.3.34 GetPlayerConnectionType()	653
6.250.3.35 GetPlayerObject()	653
6.250.3.36 GetPlayerRtt()	653
6.250.3.37 GetPlayerUserId()	654
6.250.3.38 GetRawInputForPlayer()	654
6.250.3.39 GetResumeSnapshotNetworkObjects()	654
6.250.3.40 GetResumeSnapshotNetworkSceneObjects()	655
6.250.3.41 GetRpcTargetStatus()	655
6.250.3.42 GetRunnerForGameObject()	655
6.250.3.43 GetRunnerForScene()	655
6.250.3.44 GetSingleton< T >()	656
6.250.3.45 HasSingleton< T >()	656
6.250.3.46 InstantiateInRunnerScene() [1/2]	656
6.250.3.47 InstantiateInRunnerScene() [2/2]	656
6.250.3.48 InstantiateInRunnerScene< T >() [1/2]	657
6.250.3.49 InstantiateInRunnerScene< T >() [2/2]	657
6.250.3.50 InvokeSceneLoadDone()	657
6.250.3.51 InvokeSceneLoadStart()	657
6.250.3.52 IsInterestedIn()	658
6.250.3.53 IsPlayerValid()	658
6.250.3.54 JoinSessionLobby()	658
6.250.3.55 LoadScene() [1/3]	659
6.250.3.56 LoadScene() [2/3]	659
6.250.3.57 LoadScene() [3/3]	660
6.250.3.58 MakeDontDestroyOnLoad()	660
6.250.3.59 MoveGameObjectToSameScene()	661
6.250.3.60 MoveGameObjectToScene()	661
6.250.3.61 MoveToRunnerScene()	661
6.250.3.62 MoveToRunnerScene< T >()	662

6.250.3.63 ObjectDelegate()	662
6.250.3.64 OnBeforeSpawned()	662
6.250.3.65 PushHostMigrationSnapshot()	662
6.250.3.66 RegisterSceneObjects()	663
6.250.3.67 ReleaseStateAuthority()	663
6.250.3.68 RemoveCallbacks()	663
6.250.3.69 RemoveGlobal()	664
6.250.3.70 RenderInternal()	664
6.250.3.71 RequestStateAuthority()	664
6.250.3.72 SendReliableDataToPlayer()	664
6.250.3.73 SendReliableDataToServer()	665
6.250.3.74 SendRpc() [1/2]	665
6.250.3.75 SendRpc() [2/2]	665
6.250.3.76 SetAreaOfInterestCellSize()	666
6.250.3.77 SetAreaOfInterestGrid()	666
6.250.3.78 SetBehaviourReplicateTo()	666
6.250.3.79 SetBehaviourReplicateToAll()	667
6.250.3.80 SetIsSimulated()	667
6.250.3.81 SetMasterClient()	667
6.250.3.82 SetPlayerAlwaysInterested()	668
6.250.3.83 SetPlayerObject()	668
6.250.3.84 SetSimulateMultiPeerPhysics()	668
6.250.3.85 Shutdown()	669
6.250.3.86 SinglePlayerContinue()	669
6.250.3.87 SinglePlayerPause() [1/2]	669
6.250.3.88 SinglePlayerPause() [2/2]	669
6.250.3.89 Spawn() [1/5]	669
6.250.3.90 Spawn() [2/5]	670
6.250.3.91 Spawn() [3/5]	670
6.250.3.92 Spawn() [4/5]	671
6.250.3.93 Spawn() [5/5]	672
6.250.3.94 Spawn< T >()	672
6.250.3.95 SpawnAsync() [1/5]	673
6.250.3.96 SpawnAsync() [2/5]	674
6.250.3.97 SpawnAsync() [3/5]	674
6.250.3.98 SpawnAsync() [4/5]	675
6.250.3.99 SpawnAsync() [5/5]	675
6.250.3.100 SpawnAsync< T >()	676
6.250.3.101 StartGame()	677
6.250.3.102 TryFindBehaviour()	677
6.250.3.103 TryFindBehaviour< T >()	678
6.250.3.104 TryFindObject()	678

6.250.3.105 TryGetBehaviourStatistics()	679
6.250.3.106 TryGetFusionStatistics()	679
6.250.3.107 TryGetInputForPlayer< T >()	680
6.250.3.108 TryGetNetworkedBehaviourFromNetworkedObjectRef< T >()	680
6.250.3.109 TryGetNetworkedBehaviourId()	680
6.250.3.110 TryGetObjectRefFromNetworkedBehaviour()	681
6.250.3.111 TryGetPhysicsInfo()	681
6.250.3.112 TryGetPlayerObject()	682
6.250.3.113 TryGetSceneInfo()	682
6.250.3.114 TrySetPhysicsInfo()	682
6.250.3.115 TrySpawn() [1/5]	683
6.250.3.116 TrySpawn() [2/5]	683
6.250.3.117 TrySpawn() [3/5]	685
6.250.3.118 TrySpawn() [4/5]	686
6.250.3.119 TrySpawn() [5/5]	686
6.250.3.120 TrySpawn< T >()	687
6.250.3.121 UnloadScene()	687
6.250.3.122 UpdateInternal()	689
6.250.4 Property Documentation	689
6.250.4.1 ActivePlayers	689
6.250.4.2 AuthenticationValues	689
6.250.4.3 BuildType	689
6.250.4.4 CanSpawn	690
6.250.4.5 Config	690
6.250.4.6 CurrentConnectionType	690
6.250.4.7 DeltaTime	690
6.250.4.8 GameMode	690
6.250.4.9 Instances	690
6.250.4.10 IsClient	691
6.250.4.11 IsCloudReady	691
6.250.4.12 IsConnectedToServer	691
6.250.4.13 IsFirstTick	691
6.250.4.14 IsForward	691
6.250.4.15 IsLastTick	691
6.250.4.16 IsPlayer	692
6.250.4.17 IsResimulation	692
6.250.4.18 IsResume	692
6.250.4.19 IsRunning	692
6.250.4.20 IsSceneAuthority	692
6.250.4.21 IsSceneManagerBusy	692
6.250.4.22 IsServer	693
6.250.4.23 IsSharedModeMasterClient	693

6.250.4.24 IsShutdown	693
6.250.4.25 IsSimulationUpdating	693
6.250.4.26 IsSinglePlayer	693
6.250.4.27 IsStarting	693
6.250.4.28 LagCompensation	694
6.250.4.29 LatestServerTick	694
6.250.4.30 LobbyInfo	694
6.250.4.31 LocalAlpha	694
6.250.4.32 LocalPlayer	694
6.250.4.33 LocalRenderTime	694
6.250.4.34 Mode	695
6.250.4.35 NATType	695
6.250.4.36 ObjectProvider	695
6.250.4.37 Prefabs	695
6.250.4.38 ProvideInput	695
6.250.4.39 RemoteRenderTime	695
6.250.4.40 SceneManager	696
6.250.4.41 SessionInfo	696
6.250.4.42 SimulationTime	696
6.250.4.43 SimulationUnityScene	696
6.250.4.44 Stage	696
6.250.4.45 State	696
6.250.4.46 Tick	697
6.250.4.47 TickRate	697
6.250.4.48 TicksExecuted	697
6.250.4.49 Topology	697
6.250.4.50 UserId	697
6.250.5 Event Documentation	697
6.250.5.1 ObjectAcquired	697
6.251 NetworkRunnerCallbackArgs Class Reference	698
6.251.1 Detailed Description	698
6.252 NetworkRunnerCallbackArgs.ConnectRequest Class Reference	698
6.252.1 Detailed Description	698
6.252.2 Member Function Documentation	698
6.252.2.1 Accept()	699
6.252.2.2 Refuse()	699
6.252.2.3 Waiting()	699
6.252.3 Property Documentation	699
6.252.3.1 RemoteAddress	699
6.253 NetworkRunnerUpdaterDefault Class Reference	699
6.253.1 Detailed Description	700
6.253.2 Member Function Documentation	700

6.253.2.1 RegisterInPlayerLoop()	700
6.253.2.2 UnregisterFromPlayerLoop()	701
6.253.3 Member Data Documentation	701
6.253.3.1 RenderSettings	701
6.253.3.2 UpdateSettings	701
6.254 NetworkRunnerUpdaterDefault.NetworkRunnerRender Struct Reference	701
6.254.1 Detailed Description	701
6.255 NetworkRunnerUpdaterDefault.NetworkRunnerUpdate Struct Reference	702
6.255.1 Detailed Description	702
6.256 NetworkRunnerUpdaterDefault.InvokeSettings Struct Reference	702
6.256.1 Detailed Description	702
6.256.2 Member Function Documentation	703
6.256.2.1 Equals() [1/2]	703
6.256.2.2 Equals() [2/2]	703
6.256.2.3 GetHashCode()	703
6.256.2.4 operator"!="()	704
6.256.2.5 operator=="()	704
6.256.2.6 ToString()	704
6.256.3 Member Data Documentation	705
6.256.3.1 AddMode	705
6.256.3.2 ReferencePlayerLoopSystem	705
6.257 NetworkSceneAsyncOp Struct Reference	705
6.257.1 Detailed Description	706
6.257.2 Member Function Documentation	706
6.257.2.1 AddOnCompleted()	706
6.257.2.2 FromAsyncOperation()	706
6.257.2.3 FromCompleted()	707
6.257.2.4 FromCoroutine()	707
6.257.2.5 FromError()	708
6.257.2.6 FromTask()	708
6.257.2.7 GetAwaiter()	709
6.257.3 Member Data Documentation	709
6.257.3.1 SceneRef	709
6.257.4 Property Documentation	709
6.257.4.1 Error	709
6.257.4.2 IsDone	709
6.257.4.3 IsValid	709
6.258 NetworkSceneAsyncOp.Awaiter Struct Reference	710
6.258.1 Detailed Description	710
6.258.2 Constructor & Destructor Documentation	710
6.258.2.1 Awaiter()	710
6.258.3 Member Function Documentation	711

6.258.3.1 GetResult()	711
6.258.3.2 OnCompleted()	711
6.258.4 Property Documentation	711
6.258.4.1 IsCompleted	711
6.259 NetworkSceneInfo Struct Reference	711
6.259.1 Detailed Description	712
6.259.2 Member Function Documentation	713
6.259.2.1 AddSceneRef()	713
6.259.2.2 Equals() [1/2]	713
6.259.2.3 Equals() [2/2]	713
6.259.2.4 GetHashCode()	714
6.259.2.5 IndexOf()	714
6.259.2.6 operator NetworkSceneInfo()	714
6.259.2.7 RemoveSceneRef()	715
6.259.2.8 ToString()	715
6.259.3 Member Data Documentation	715
6.259.3.1 MaxScenes	715
6.259.3.2 SIZE	716
6.259.3.3 WORD_COUNT	716
6.259.4 Property Documentation	716
6.259.4.1 SceneCount	716
6.259.4.2 SceneParams	716
6.259.4.3 Scenes	716
6.259.4.4 Version	716
6.260 NetworkSceneLoadId Struct Reference	717
6.260.1 Detailed Description	717
6.260.2 Constructor & Destructor Documentation	717
6.260.2.1 NetworkSceneLoadId()	717
6.260.3 Member Function Documentation	718
6.260.3.1 Equals() [1/2]	718
6.260.3.2 Equals() [2/2]	718
6.260.3.3 GetHashCode()	718
6.260.3.4 operator NetworkSceneLoadId()	719
6.260.3.5 operator"!=()	719
6.260.3.6 operator"==()	719
6.260.3.7 ToString()	720
6.260.4 Member Data Documentation	720
6.260.4.1 Value	720
6.261 NetworkSceneObjectId Struct Reference	720
6.261.1 Detailed Description	721
6.261.2 Constructor & Destructor Documentation	721
6.261.2.1 NetworkSceneObjectId()	721

6.261.3 Member Function Documentation	721
6.261.3.1 Equals() [1/2]	721
6.261.3.2 Equals() [2/2]	722
6.261.3.3 GetHashCode()	722
6.261.3.4 ToString()	722
6.261.4 Member Data Documentation	722
6.261.4.1 LoadId	722
6.261.4.2 ObjectId	723
6.261.4.3 Scene	723
6.261.5 Property Documentation	723
6.261.5.1 IsValid	723
6.261.5.2 SceneLoadId	723
6.262 NetworkSerializeMethodAttribute Class Reference	723
6.262.1 Detailed Description	724
6.262.2 Property Documentation	724
6.262.2.1 MaxSize	724
6.263 NetworkSimulationConfiguration Class Reference	724
6.263.1 Detailed Description	725
6.263.2 Member Function Documentation	725
6.263.2.1 Clone()	725
6.263.2.2 Create()	725
6.263.3 Member Data Documentation	725
6.263.3.1 AdditionalJitter	726
6.263.3.2 AdditionalLoss	726
6.263.3.3 DelayMax	726
6.263.3.4 DelayMin	726
6.263.3.5 DelayPeriod	726
6.263.3.6 DelayShape	726
6.263.3.7 DelayThreshold	727
6.263.3.8 Enabled	727
6.263.3.9 LossChanceMax	727
6.263.3.10 LossChanceMin	727
6.263.3.11 LossChancePeriod	727
6.263.3.12 LossChanceShape	727
6.263.3.13 LossChanceThreshold	728
6.264 NetworkSpawnOp Struct Reference	728
6.264.1 Detailed Description	728
6.264.2 Member Function Documentation	729
6.264.2.1 GetAwaiter()	729
6.264.3 Member Data Documentation	729
6.264.3.1 Runner	729
6.264.4 Property Documentation	729

6.264.4.1 IsFailed	729
6.264.4.2 IsQueued	729
6.264.4.3 IsSpawned	729
6.264.4.4 Object	730
6.264.4.5 Status	730
6.265 NetworkSpawnOp.Awaiter Struct Reference	730
6.265.1 Detailed Description	730
6.265.2 Constructor & Destructor Documentation	730
6.265.2.1 Awaiter()	730
6.265.3 Member Function Documentation	731
6.265.3.1 GetResult()	731
6.265.3.2 OnCompleted()	731
6.265.4 Property Documentation	731
6.265.4.1 IsCompleted	732
6.266 NetworkString< TSize > Class Template Reference	732
6.266.1 Detailed Description	734
6.266.2 Constructor & Destructor Documentation	735
6.266.2.1 NetworkString()	735
6.266.3 Member Function Documentation	735
6.266.3.1 Assign()	735
6.266.3.2 Compare() [1/3]	735
6.266.3.3 Compare() [2/3]	736
6.266.3.4 Compare() [3/3]	736
6.266.3.5 Compare< TOtherSize >() [1/2]	736
6.266.3.6 Compare< TOtherSize >() [2/2]	737
6.266.3.7 Contains() [1/3]	738
6.266.3.8 Contains() [2/3]	738
6.266.3.9 Contains() [3/3]	738
6.266.3.10 Contains< TOtherSize >() [1/2]	739
6.266.3.11 Contains< TOtherSize >() [2/2]	739
6.266.3.12 EndsWith()	740
6.266.3.13 EndsWith< TOtherSize >()	740
6.266.3.14 Equals() [1/4]	741
6.266.3.15 Equals() [2/4]	741
6.266.3.16 Equals() [3/4]	742
6.266.3.17 Equals() [4/4]	742
6.266.3.18 Equals< TOtherSize >() [1/2]	742
6.266.3.19 Equals< TOtherSize >() [2/2]	743
6.266.3.20 Get()	743
6.266.3.21 GetCapacity< TSize >()	744
6.266.3.22 GetCharCount()	744
6.266.3.23 GetEnumerator()	744

6.266.3.24 GetHashCode()	745
6.266.3.25 IndexOf() [1/6]	745
6.266.3.26 IndexOf() [2/6]	745
6.266.3.27 IndexOf() [3/6]	746
6.266.3.28 IndexOf() [4/6]	746
6.266.3.29 IndexOf() [5/6]	747
6.266.3.30 IndexOf() [6/6]	747
6.266.3.31 IndexOf< TOtherSize >() [1/4]	748
6.266.3.32 IndexOf< TOtherSize >() [2/4]	748
6.266.3.33 IndexOf< TOtherSize >() [3/4]	749
6.266.3.34 IndexOf< TOtherSize >() [4/4]	750
6.266.3.35 operator NetworkString< TSize >()	751
6.266.3.36 operator string()	751
6.266.3.37 operator"!="" [1/3]	751
6.266.3.38 operator"!="" [2/3]	752
6.266.3.39 operator"!="" [3/3]	752
6.266.3.40 operator==" [1/3]	752
6.266.3.41 operator==" [2/3]	753
6.266.3.42 operator==" [3/3]	753
6.266.3.43 Set()	754
6.266.3.44 StartsWith()	754
6.266.3.45 StartsWith< TOtherSize >()	754
6.266.3.46 Substring() [1/2]	755
6.266.3.47 Substring() [2/2]	755
6.266.3.48 ToLower()	756
6.266.3.49 ToString()	756
6.266.3.50 ToUpper()	756
6.266.4 Property Documentation	757
6.266.4.1 Capacity	757
6.266.4.2 Length	757
6.266.4.3 this[int index]	757
6.266.4.4 Value	757
6.267 NetworkStructUtils Class Reference	757
6.267.1 Detailed Description	758
6.267.2 Member Function Documentation	758
6.267.2.1 GetWordCount< T >()	758
6.268 NetworkStructWeavedAttribute Class Reference	758
6.268.1 Detailed Description	759
6.268.2 Constructor & Destructor Documentation	759
6.268.2.1 NetworkStructWeavedAttribute()	759
6.268.3 Property Documentation	759
6.268.3.1 WordCount	759

6.269 NetworkTransform Class Reference	759
6.269.1 Detailed Description	760
6.269.2 Member Function Documentation	760
6.269.2.1 SetAreaOfInterestOverride()	760
6.269.2.2 Teleport()	761
6.269.3 Member Data Documentation	761
6.269.3.1 DisableSharedModelInterpolation	761
6.269.3.2 SyncParent	761
6.269.3.3 SyncScale	761
6.269.4 Property Documentation	762
6.269.4.1 AutoUpdateAreaOfInterestOverride	762
6.270 NetworkTRSP Class Reference	762
6.270.1 Detailed Description	763
6.270.2 Member Function Documentation	763
6.270.2.1 Render()	763
6.270.2.2 ResolveAOIOVERRIDE()	763
6.270.2.3 SetAreaOfInterestOverride()	764
6.270.2.4 SetParentTransform()	764
6.270.2.5 Teleport()	764
6.270.3 Member Data Documentation	764
6.270.3.1 reenabledTick	765
6.270.4 Property Documentation	765
6.270.4.1 Data	765
6.270.4.2 IsMainTRSP	765
6.270.4.3 State	765
6.271 NetworkTRSPData Struct Reference	765
6.271.1 Detailed Description	766
6.271.2 Member Data Documentation	766
6.271.2.1 AreaOfInterestOverride	766
6.271.2.2 Parent	767
6.271.2.3 Position	767
6.271.2.4 POSITION_OFFSET	767
6.271.2.5 Rotation	767
6.271.2.6 Scale	767
6.271.2.7 SIZE	767
6.271.2.8 TeleportKey	768
6.271.2.9 WORDS	768
6.271.3 Property Documentation	768
6.271.3.1 NonNetworkedParent	768
6.272 NormalizedRectAttribute Class Reference	768
6.272.1 Detailed Description	769
6.272.2 Constructor & Destructor Documentation	769

6.272.2.1 NormalizedRectAttribute()	769
6.272.3 Member Data Documentation	769
6.272.3.1 AspectRatio	769
6.272.3.2 InvertY	769
6.273 OnChangedRenderAttribute Class Reference	769
6.273.1 Detailed Description	770
6.273.2 Constructor & Destructor Documentation	770
6.273.2.1 OnChangedRenderAttribute()	770
6.273.3 Property Documentation	770
6.273.3.1 MethodName	770
6.274 PlayerRef Struct Reference	771
6.274.1 Detailed Description	772
6.274.2 Member Function Documentation	772
6.274.2.1 Equals() [1/2]	772
6.274.2.2 Equals() [2/2]	773
6.274.2.3 FromEncoded()	773
6.274.2.4 FromIndex()	773
6.274.2.5 GetHashCode()	775
6.274.2.6 operator"!=()	775
6.274.2.7 operator=="()	775
6.274.2.8 Read()	776
6.274.2.9 ToString()	776
6.274.2.10 Write()	776
6.274.2.11 Write< T >()	776
6.274.3 Member Data Documentation	777
6.274.3.1 MASTER_CLIENT_RAW	777
6.274.3.2 SIZE	777
6.274.4 Property Documentation	777
6.274.4.1 AsIndex	777
6.274.4.2 Comparer	778
6.274.4.3 Invalid	778
6.274.4.4 IsMasterClient	778
6.274.4.5 IsNone	778
6.274.4.6 IsRealPlayer	778
6.274.4.7 MasterClient	778
6.274.4.8 None	779
6.274.4.9 PlayerId	779
6.274.4.10 RawEncoded	779
6.275 PreserveInPluginAttribute Class Reference	779
6.275.1 Detailed Description	779
6.275.2 Constructor & Destructor Documentation	779
6.275.2.1 PreserveInPluginAttribute()	780

6.275.3 Property Documentation	780
6.275.3.1 KeepNonStateMembers	780
6.276 Primes Class Reference	780
6.276.1 Detailed Description	780
6.276.2 Member Function Documentation	780
6.276.2.1 GetNextPrime() [1/2]	780
6.276.2.2 GetNextPrime() [2/2]	781
6.276.2.3 IsPrime()	781
6.277 PropertyAttribute Class Reference	782
6.277.1 Detailed Description	782
6.278 BitStream Class Reference	782
6.278.1 Detailed Description	787
6.278.2 Constructor & Destructor Documentation	787
6.278.2.1 BitStream() [1/4]	787
6.278.2.2 BitStream() [2/4]	787
6.278.2.3 BitStream() [3/4]	788
6.278.2.4 BitStream() [4/4]	788
6.278.3 Member Function Documentation	788
6.278.3.1 ArrayElementSerializer< T >()	788
6.278.3.2 CanRead() [1/2]	789
6.278.3.3 CanRead() [2/2]	789
6.278.3.4 CanWrite() [1/2]	789
6.278.3.5 CanWrite() [2/2]	789
6.278.3.6 Condition()	790
6.278.3.7 CopyFromArray()	790
6.278.3.8 Expand()	790
6.278.3.9 ReadBool()	791
6.278.3.10 ReadBoolean()	791
6.278.3.11 ReadByte() [1/2]	791
6.278.3.12 ReadByte() [2/2]	791
6.278.3.13 ReadByteArray() [1/4]	792
6.278.3.14 ReadByteArray() [2/4]	792
6.278.3.15 ReadByteArray() [3/4]	792
6.278.3.16 ReadByteArray() [4/4]	793
6.278.3.17 ReadByteArrayLengthPrefixed()	793
6.278.3.18 ReadChar()	793
6.278.3.19 ReadDouble()	794
6.278.3.20 ReadFloat()	794
6.278.3.21 ReadGuid()	794
6.278.3.22 ReadInt() [1/2]	794
6.278.3.23 ReadInt() [2/2]	794
6.278.3.24 ReadInt_Shifted()	795

6.278.3.25 ReadLong() [1/2]	795
6.278.3.26 ReadLong() [2/2]	795
6.278.3.27 ReadSByte()	796
6.278.3.28 ReadShort() [1/2]	796
6.278.3.29 ReadShort() [2/2]	796
6.278.3.30 ReadString() [1/2]	797
6.278.3.31 ReadString() [2/2]	797
6.278.3.32 ReadUInt() [1/2]	797
6.278.3.33 ReadUInt() [2/2]	797
6.278.3.34 ReadULong() [1/2]	798
6.278.3.35 ReadULong() [2/2]	798
6.278.3.36 ReadUShort() [1/2]	798
6.278.3.37 ReadUShort() [2/2]	799
6.278.3.38 Reset() [1/2]	799
6.278.3.39 Reset() [2/2]	799
6.278.3.40 ResetFast()	799
6.278.3.41 RoundToByte()	800
6.278.3.42 Serialize() [1/27]	800
6.278.3.43 Serialize() [2/27]	800
6.278.3.44 Serialize() [3/27]	800
6.278.3.45 Serialize() [4/27]	801
6.278.3.46 Serialize() [5/27]	801
6.278.3.47 Serialize() [6/27]	801
6.278.3.48 Serialize() [7/27]	802
6.278.3.49 Serialize() [8/27]	802
6.278.3.50 Serialize() [9/27]	802
6.278.3.51 Serialize() [10/27]	802
6.278.3.52 Serialize() [11/27]	803
6.278.3.53 Serialize() [12/27]	803
6.278.3.54 Serialize() [13/27]	803
6.278.3.55 Serialize() [14/27]	804
6.278.3.56 Serialize() [15/27]	804
6.278.3.57 Serialize() [16/27]	804
6.278.3.58 Serialize() [17/27]	804
6.278.3.59 Serialize() [18/27]	806
6.278.3.60 Serialize() [19/27]	806
6.278.3.61 Serialize() [20/27]	806
6.278.3.62 Serialize() [21/27]	807
6.278.3.63 Serialize() [22/27]	807
6.278.3.64 Serialize() [23/27]	807
6.278.3.65 Serialize() [24/27]	807
6.278.3.66 Serialize() [25/27]	808

6.278.3.67 Serialize() [26/27]	808
6.278.3.68 Serialize() [27/27]	808
6.278.3.69 SerializeArray< T >()	809
6.278.3.70 SerializeArrayLength< T >()	809
6.278.3.71 SerializeBuffer() [1/8]	809
6.278.3.72 SerializeBuffer() [2/8]	810
6.278.3.73 SerializeBuffer() [3/8]	810
6.278.3.74 SerializeBuffer() [4/8]	810
6.278.3.75 SerializeBuffer() [5/8]	811
6.278.3.76 SerializeBuffer() [6/8]	811
6.278.3.77 SerializeBuffer() [7/8]	811
6.278.3.78 SerializeBuffer() [8/8]	811
6.278.3.79 SetBuffer() [1/2]	812
6.278.3.80 SetBuffer() [2/2]	812
6.278.3.81 ToArray()	812
6.278.3.82 WriteBool()	813
6.278.3.83 WriteBoolean()	813
6.278.3.84 WriteByte() [1/2]	813
6.278.3.85 WriteByte() [2/2]	814
6.278.3.86 WriteByteArray() [1/3]	814
6.278.3.87 WriteByteArray() [2/3]	814
6.278.3.88 WriteByteArray() [3/3]	814
6.278.3.89 WriteByteArrayLengthPrefixed() [1/2]	815
6.278.3.90 WriteByteArrayLengthPrefixed() [2/2]	815
6.278.3.91 WriteByteAt()	815
6.278.3.92 WriteChar()	817
6.278.3.93 WriteDouble()	817
6.278.3.94 WriteFloat()	817
6.278.3.95 WriteGuid()	818
6.278.3.96 WriteInt() [1/2]	818
6.278.3.97 WriteInt() [2/2]	818
6.278.3.98 WriteInt_Shifted()	818
6.278.3.99 WriteLong() [1/2]	819
6.278.3.100 WriteLong() [2/2]	819
6.278.3.101 WriteSByte()	819
6.278.3.102 WriteShort() [1/2]	820
6.278.3.103 WriteShort() [2/2]	820
6.278.3.104 WriteString() [1/2]	820
6.278.3.105 WriteString() [2/2]	820
6.278.3.106 WriteUInt() [1/2]	821
6.278.3.107 WriteUInt() [2/2]	821
6.278.3.108 WriteULong() [1/2]	821

6.278.3.109 WriteULong() [2/2]	822
6.278.3.110 WriteUShort() [1/2]	822
6.278.3.111 WriteUShort() [2/2]	822
6.278.4 Property Documentation	822
6.278.4.1 BytesRequired	823
6.278.4.2 Capacity	823
6.278.4.3 Data	823
6.278.4.4 Done	823
6.278.4.5 IsEvenBytes	823
6.278.4.6 Overflowing	823
6.278.4.7 Position	824
6.278.4.8 Reading	824
6.278.4.9 Size	824
6.278.4.10 Writing	824
6.279 ICommunicator Interface Reference	824
6.279.1 Detailed Description	825
6.279.2 Member Function Documentation	825
6.279.2.1 Poll()	825
6.279.2.2 PushPackage()	825
6.279.2.3 ReceivePackage()	825
6.279.2.4 RegisterPackageCallback< T >()	826
6.279.2.5 SendMessage()	826
6.279.2.6 SendPackage()	827
6.279.2.7 Service()	827
6.279.3 Property Documentation	827
6.279.3.1 CommunicatorID	827
6.280 IMessage Interface Reference	827
6.280.1 Detailed Description	828
6.281 Ptr Struct Reference	828
6.281.1 Detailed Description	829
6.281.2 Member Function Documentation	829
6.281.2.1 Equals() [1/2]	829
6.281.2.2 Equals() [2/2]	829
6.281.2.3 GetHashCode()	830
6.281.2.4 operator bool()	830
6.281.2.5 operator"!="()	830
6.281.2.6 operator+()	831
6.281.2.7 operator-()	831
6.281.2.8 operator==(())	831
6.281.2.9 ToString()	832
6.281.3 Member Data Documentation	832
6.281.3.1 Address	832

6.281.3.2 SIZE	832
6.281.4 Property Documentation	832
6.281.4.1 Null	833
6.282 Ptr.EqualityComparer Class Reference	833
6.282.1 Detailed Description	833
6.282.2 Member Function Documentation	833
6.282.2.1 Equals()	833
6.282.2.2 GetHashCode()	834
6.283 QuaternionCompressed Struct Reference	834
6.283.1 Detailed Description	835
6.283.2 Member Function Documentation	835
6.283.2.1 Equals() [1/2]	835
6.283.2.2 Equals() [2/2]	835
6.283.2.3 GetHashCode()	836
6.283.2.4 operator Quaternion()	836
6.283.2.5 operator QuaternionCompressed()	836
6.283.2.6 operator "!="()	837
6.283.2.7 operator "=="()	837
6.283.3 Member Data Documentation	838
6.283.3.1 wEncoded	838
6.283.3.2 xEncoded	838
6.283.3.3 yEncoded	838
6.283.3.4 zEncoded	838
6.283.4 Property Documentation	838
6.283.4.1 W	838
6.283.4.2 X	839
6.283.4.3 Y	839
6.283.4.4 Z	839
6.284 RangeExAttribute Class Reference	839
6.284.1 Detailed Description	840
6.284.2 Constructor & Destructor Documentation	840
6.284.2.1 RangeExAttribute()	840
6.284.3 Member Data Documentation	840
6.284.3.1 ClampMax	840
6.284.3.2 ClampMin	840
6.284.3.3 UseSlider	840
6.284.4 Property Documentation	841
6.284.4.1 Max	841
6.284.4.2 Min	841
6.285 ReadOnlyAttribute Class Reference	841
6.285.1 Detailed Description	841
6.285.2 Property Documentation	841

6.285.2.1 InEditMode	842
6.285.2.2 InPlayMode	842
6.286 ReadWriteUtils Class Reference	842
6.286.1 Detailed Description	843
6.286.2 Member Function Documentation	843
6.286.2.1 ReadFloat()	843
6.286.2.2 ReadQuaternion()	843
6.286.2.3 ReadVector2()	843
6.286.2.4 ReadVector3()	844
6.286.2.5 ReadVector4()	844
6.286.2.6 WriteFloat()	844
6.286.2.7 WriteQuaternion()	845
6.286.2.8 WriteVector2()	845
6.286.2.9 WriteVector3()	845
6.286.2.10 WriteVector4()	846
6.286.3 Member Data Documentation	846
6.286.3.1 ACCURACY	846
6.287 ReadWriteUtilsForWeaver Class Reference	846
6.287.1 Detailed Description	847
6.287.2 Member Function Documentation	847
6.287.2.1 GetByteArrayHashCode()	847
6.287.2.2 GetByteCountUtf8NoHash()	848
6.287.2.3 GetStringHashCode()	848
6.287.2.4 GetWordCountString()	848
6.287.2.5 ReadBoolean()	849
6.287.2.6 ReadStringUtf32NoHash()	849
6.287.2.7 ReadStringUtf32WithHash()	849
6.287.2.8 ReadStringUtf8NoHash()	851
6.287.2.9 VerifyRawNetworkUnwrap< T >()	851
6.287.2.10 VerifyRawNetworkWrap< T >()	852
6.287.2.11 WriteBoolean()	852
6.287.2.12 WriteStringUtf32NoHash()	853
6.287.2.13 WriteStringUtf32WithHash()	853
6.287.2.14 WriteStringUtf8NoHash()	853
6.288 ReflectionUtils Class Reference	854
6.288.1 Detailed Description	854
6.288.2 Member Function Documentation	854
6.288.2.1 GetAllNetworkBehaviourTypes()	855
6.288.2.2 GetAllSimulationBehaviourTypes()	855
6.288.2.3 GetAllWeavedAssemblies()	855
6.288.2.4 GetAllWeavedNetworkBehaviourTypes()	855
6.288.2.5 GetAllWeavedSimulationBehaviourTypes()	856

6.288.2.6 GetAllWeaverGeneratedTypes()	856
6.288.2.7 GetCustomAttributeOrThrow< T >()	856
6.288.2.8 GetWeavedAttributeOrThrow()	857
6.289 RenderAttribute Class Reference	857
6.289.1 Detailed Description	858
6.289.2 Constructor & Destructor Documentation	858
6.289.2.1 RenderAttribute() [1/2]	858
6.289.2.2 RenderAttribute() [2/2]	858
6.289.3 Property Documentation	858
6.289.3.1 Method	858
6.289.3.2 Source	858
6.289.3.3 Timeframe	859
6.290 RenderTimeline Struct Reference	859
6.290.1 Detailed Description	859
6.290.2 Member Function Documentation	859
6.290.2.1 GetRenderBuffers()	859
6.291 RenderWeavedAttribute Class Reference	860
6.291.1 Detailed Description	860
6.291.2 Constructor & Destructor Documentation	860
6.291.2.1 RenderWeavedAttribute()	860
6.292 ResolveNetworkPrefabSourceAttribute Class Reference	860
6.292.1 Detailed Description	860
6.293 RpcAttribute Class Reference	860
6.293.1 Detailed Description	861
6.293.2 Constructor & Destructor Documentation	862
6.293.2.1 RpcAttribute() [1/2]	862
6.293.2.2 RpcAttribute() [2/2]	862
6.293.3 Member Data Documentation	862
6.293.3.1 MaxPayloadSize	862
6.293.4 Property Documentation	862
6.293.4.1 Channel	862
6.293.4.2 HostMode	863
6.293.4.3 InvokeLocal	863
6.293.4.4 Sources	863
6.293.4.5 Targets	863
6.293.4.6 TickAligned	863
6.294 RpcHeader Struct Reference	863
6.294.1 Detailed Description	864
6.294.2 Member Function Documentation	864
6.294.2.1 Create() [1/2]	864
6.294.2.2 Create() [2/2]	865
6.294.2.3 Read()	865

6.294.2.4 ReadSize()	866
6.294.2.5 ToString()	866
6.294.2.6 Write()	866
6.294.3 Member Data Documentation	866
6.294.3.1 Behaviour	867
6.294.3.2 Method	867
6.294.3.3 Object	867
6.294.3.4 SIZE	867
6.295 RpcInfo Struct Reference	867
6.295.1 Detailed Description	868
6.295.2 Member Function Documentation	868
6.295.2.1 FromLocal()	868
6.295.2.2 FromMessage()	868
6.295.2.3 ToString()	869
6.295.3 Member Data Documentation	869
6.295.3.1 Channel	869
6.295.3.2 IsInvokeLocal	869
6.295.3.3 Source	869
6.295.3.4 Tick	870
6.296 RpcInvokeData Struct Reference	870
6.296.1 Detailed Description	870
6.296.2 Member Function Documentation	870
6.296.2.1 ToString()	870
6.296.3 Member Data Documentation	871
6.296.3.1 Delegate	871
6.296.3.2 Key	871
6.296.3.3 Sources	871
6.296.3.4 Targets	871
6.297 RpcInvokeInfo Struct Reference	871
6.297.1 Detailed Description	872
6.297.2 Member Function Documentation	872
6.297.2.1 ToString()	872
6.297.3 Member Data Documentation	872
6.297.3.1 LocalInvokeResult	872
6.297.3.2 SendCullResult	872
6.297.3.3 SendResult	873
6.298 RpcSendResult Struct Reference	873
6.298.1 Detailed Description	873
6.298.2 Member Function Documentation	873
6.298.2.1 ToString()	873
6.298.3 Member Data Documentation	873
6.298.3.1 MessageSize	874

6.298.3.2 Result	874
6.299 RpcTargetAttribute Class Reference	874
6.299.1 Detailed Description	874
6.299.2 Constructor & Destructor Documentation	874
6.299.2.1 RpcTargetAttribute()	874
6.300 RuntimeUnityFlagsSetup Class Reference	875
6.300.1 Detailed Description	875
6.300.2 Member Function Documentation	875
6.300.2.1 Check_ENABLE_IL2CPP()	875
6.300.2.2 Check_ENABLE_MONO()	876
6.300.2.3 Check_NET_4_6()	876
6.300.2.4 Check_NET_STANDARD_2_0()	876
6.300.2.5 Check_NETFX_CORE()	876
6.300.2.6 Check_UNITY_2019_4_OR_NEWER()	876
6.300.2.7 Check_UNITY_EDITOR()	876
6.300.2.8 Check_UNITY_GAMECORE()	877
6.300.2.9 Check_UNITY_SWITCH()	877
6.300.2.10 Check_UNITY_WEBGL()	877
6.300.2.11 Check_UNITY_XBOXONE()	877
6.300.2.12 Reset()	877
6.301 SceneLoadDoneArgs Struct Reference	877
6.301.1 Detailed Description	878
6.301.2 Constructor & Destructor Documentation	878
6.301.2.1 SceneLoadDoneArgs()	878
6.301.3 Member Data Documentation	878
6.301.3.1 RootGameObjects	878
6.301.3.2 Scene	879
6.301.3.3 SceneObjects	879
6.301.3.4 SceneRef	879
6.302 ScenePathAttribute Class Reference	879
6.302.1 Detailed Description	879
6.303 SceneRef Struct Reference	879
6.303.1 Detailed Description	881
6.303.2 Member Function Documentation	881
6.303.2.1 Equals() [1/2]	881
6.303.2.2 Equals() [2/2]	881
6.303.2.3 FromIndex()	882
6.303.2.4 FromPath()	882
6.303.2.5 FromRaw()	883
6.303.2.6 GetHashCode()	883
6.303.2.7 IsPath()	883
6.303.2.8 operator"!=()	884

6.303.2.9 operator==()	884
6.303.2.10 Parse()	884
6.303.2.11 ToString() [1/2]	885
6.303.2.12 ToString() [2/2]	885
6.303.3 Member Data Documentation	885
6.303.3.1 FLAG_ADDRESSABLE	885
6.303.3.2 RawValue	886
6.303.3.3 SIZE	886
6.303.4 Property Documentation	886
6.303.4.1 AsIndex	886
6.303.4.2 AsPathHash	886
6.303.4.3 IsIndex	886
6.303.4.4 IsValid	887
6.303.4.5 None	887
6.304 ScriptHelpAttribute Class Reference	887
6.304.1 Detailed Description	887
6.304.2 Property Documentation	887
6.304.2.1 BackColor	887
6.304.2.2 Hide	888
6.304.2.3 Style	888
6.304.2.4 Url	888
6.305 SerializableDictionary< TKey, TValue > Class Template Reference	888
6.305.1 Detailed Description	889
6.305.2 Member Function Documentation	889
6.305.2.1 Add()	889
6.305.2.2 Clear()	891
6.305.2.3 ContainsKey()	891
6.305.2.4 Create< TKey, TValue >()	891
6.305.2.5 GetEnumerator()	892
6.305.2.6 Remove()	892
6.305.2.7 Reset()	892
6.305.2.8 Store()	892
6.305.2.9 TryGetValue()	892
6.305.2.10 Wrap()	893
6.305.3 Member Data Documentation	893
6.305.3.1 EntryKeyPropertyPath	893
6.305.3.2 ItemsPropertyPath	893
6.305.4 Property Documentation	894
6.305.4.1 Count	894
6.305.4.2 IsReadOnly	894
6.305.4.3 Keys	894
6.305.4.4 this[TKey key]	894

6.305.4.5 Values	895
6.306 SerializableType< BaseType > Struct Template Reference	895
6.306.1 Detailed Description	896
6.306.2 Constructor & Destructor Documentation	896
6.306.2.1 SerializableType() [1/2]	896
6.306.2.2 SerializableType() [2/2]	896
6.306.3 Member Function Documentation	897
6.306.3.1 AsShort()	897
6.306.3.2 Equals() [1/2]	897
6.306.3.3 Equals() [2/2]	897
6.306.3.4 GetHashCode()	897
6.306.3.5 GetShortAssemblyQualifiedName()	897
6.306.3.6 operator SerializableType()	898
6.306.3.7 operator Type()	898
6.306.4 Member Data Documentation	898
6.306.4.1 AssemblyQualifiedName	898
6.306.5 Property Documentation	898
6.306.5.1 IsValid	898
6.306.5.2 Value	898
6.307 SerializableTypeAttribute Class Reference	899
6.307.1 Detailed Description	899
6.307.2 Property Documentation	899
6.307.2.1 BaseType	899
6.307.2.2 UseFullAssemblyQualifiedName	899
6.307.2.3 WarnIfNoPreserveAttribute	899
6.308 SerializeReferenceTypePickerAttribute Class Reference	900
6.308.1 Detailed Description	900
6.308.2 Constructor & Destructor Documentation	900
6.308.2.1 SerializeReferenceTypePickerAttribute()	900
6.308.3 Member Data Documentation	901
6.308.3.1 GroupTypesByNamespace	901
6.308.3.2 ShowFullName	901
6.308.4 Property Documentation	901
6.308.4.1 Types	901
6.309 SessionInfo Class Reference	901
6.309.1 Detailed Description	902
6.309.2 Member Function Documentation	902
6.309.2.1 operator bool()	902
6.309.2.2 ToString()	902
6.309.2.3 UpdateCustomProperties()	903
6.309.3 Property Documentation	903
6.309.3.1 IsOpen	903

6.309.3.2 IsValid	903
6.309.3.3 IsVisible	903
6.309.3.4 MaxPlayers	904
6.309.3.5 Name	904
6.309.3.6 PlayerCount	904
6.309.3.7 Properties	904
6.309.3.8 Region	904
6.310 Simulation Class Reference	904
6.310.1 Detailed Description	907
6.310.2 Member Function Documentation	908
6.310.2.1 AfterSimulation()	908
6.310.2.2 AfterUpdate()	908
6.310.2.3 BeforeFirstTick()	908
6.310.2.4 BeforeSimulation()	908
6.310.2.5 BeforeUpdate()	908
6.310.2.6 GetAreaOfInterestGizmoData()	908
6.310.2.7 GetInputAuthority()	909
6.310.2.8 GetInputForPlayer()	909
6.310.2.9 GetObjectsAndPlayersInAreaOfInterestCell()	909
6.310.2.10 GetObjectsInAreaOfInterestForPlayer()	910
6.310.2.11 GetStateAuthority()	910
6.310.2.12 HasAnyActiveConnections()	910
6.310.2.13 IsInputAuthority()	910
6.310.2.14 IsInterestedIn()	911
6.310.2.15 IsLocalSimulationInputAuthority()	911
6.310.2.16 IsLocalSimulationStateAuthority() [1/2]	912
6.310.2.17 IsLocalSimulationStateAuthority() [2/2]	912
6.310.2.18 IsLocalSimulationStateOrInputSource()	912
6.310.2.19 IsStateAuthority() [1/2]	913
6.310.2.20 IsStateAuthority() [2/2]	913
6.310.2.21 NetworkConnected()	913
6.310.2.22 NetworkDisconnected()	914
6.310.2.23 NetworkReceiveDone()	914
6.310.2.24 NoSimulation()	914
6.310.2.25 TryGetHostPlayer()	914
6.310.2.26 Update()	915
6.310.3 Property Documentation	915
6.310.3.1 ActivePlayers	915
6.310.3.2 Config	915
6.310.3.3 DeltaTime	915
6.310.3.4 InputCount	916
6.310.3.5 IsClient	916

6.310.3.6 IsFirstTick	916
6.310.3.7 IsForward	916
6.310.3.8 IsLastTick	916
6.310.3.9 IsLocalPlayerFirstExecution	917
6.310.3.10 IsMasterClient	917
6.310.3.11 IsPlayer	917
6.310.3.12 IsResimulation	917
6.310.3.13 IsRunning	917
6.310.3.14 IsServer	917
6.310.3.15 IsShutdown	918
6.310.3.16 IsSinglePlayer	918
6.310.3.17 LatestServerTick	918
6.310.3.18 LocalAddress	918
6.310.3.19 LocalAlpha	918
6.310.3.20 LocalPlayer	918
6.310.3.21 Mode	919
6.310.3.22 NetConfigPointer	919
6.310.3.23 ObjectCount	919
6.310.3.24 Objects	919
6.310.3.25 ProjectConfig	919
6.310.3.26 RemoteAlpha	919
6.310.3.27 RemoteTick	920
6.310.3.28 RemoteTickPrevious	920
6.310.3.29 SendDelta	920
6.310.3.30 SendRate	920
6.310.3.31 Stage	920
6.310.3.32 Tick	920
6.310.3.33 TickDeltaDouble	921
6.310.3.34 TickDeltaFloat	921
6.310.3.35 TickPrevious	921
6.310.3.36 TickRate	921
6.310.3.37 TickStride	921
6.310.3.38 Time	921
6.310.3.39 Topology	922
6.311 Simulation.AreaOfInterest Struct Reference	922
6.311.1 Detailed Description	922
6.311.2 Member Function Documentation	922
6.311.2.1 GetCellSize()	923
6.311.2.2 SphereToCells()	923
6.311.2.3 ToCell() [1/2]	923
6.311.2.4 ToCell() [2/2]	924
6.311.2.5 ToCellCenter()	924

6.311.3 Member Data Documentation	924
6.311.3.1 CELL_SIZE	924
6.311.3.2 x	925
6.312 SimulationBehaviour Class Reference	925
6.312.1 Detailed Description	926
6.312.2 Member Function Documentation	926
6.312.2.1 FixedUpdateNetwork()	926
6.312.2.2 Render()	926
6.312.3 Property Documentation	927
6.312.3.1 CanReceiveRenderCallback	927
6.312.3.2 CanReceiveSimulationCallback	927
6.312.3.3 Object	927
6.312.3.4 Runner	927
6.313 SimulationBehaviourAttribute Class Reference	927
6.313.1 Detailed Description	928
6.313.2 Property Documentation	928
6.313.2.1 Modes	928
6.313.2.2 Stages	928
6.313.2.3 Topologies	928
6.314 SimulationBehaviourListScope Struct Reference	929
6.314.1 Detailed Description	929
6.314.2 Member Function Documentation	929
6.314.2.1 Dispose()	929
6.315 SimulationConfig Class Reference	929
6.315.1 Detailed Description	930
6.315.2 Member Enumeration Documentation	930
6.315.2.1 DataConsistency	930
6.315.2.2 InputTransferModes	931
6.315.2.3 SimulationTimeMode	931
6.315.3 Member Data Documentation	931
6.315.3.1 HostMigration	931
6.315.3.2 InputDataWordCount	932
6.315.3.3 InputTransferMode	932
6.315.3.4 ObjectDataConsistency	932
6.315.3.5 PlayerCount	932
6.315.3.6 ReplicationFeatures	932
6.315.3.7 SimulationUpdateTimeMode	932
6.315.3.8 TickRateSelection	933
6.315.3.9 Topology	933
6.315.4 Property Documentation	933
6.315.4.1 AreaOfInterestEnabled	933
6.315.4.2 InputTotalWordCount	933

6.315.4.3 SchedulingEnabled	933
6.315.4.4 SchedulingWithoutAOI	933
6.316 SimulationInput Class Reference	934
6.316.1 Detailed Description	934
6.316.2 Member Function Documentation	934
6.316.2.1 Clear()	934
6.316.2.2 CopyFrom()	935
6.316.3 Property Documentation	935
6.316.3.1 Data	935
6.316.3.2 Header	935
6.316.3.3 Player	935
6.316.3.4 Sent	935
6.317 SimulationInput.Buffer Class Reference	936
6.317.1 Detailed Description	936
6.317.2 Constructor & Destructor Documentation	936
6.317.2.1 Buffer()	936
6.317.3 Member Function Documentation	937
6.317.3.1 Add()	937
6.317.3.2 Clear()	937
6.317.3.3 Contains()	937
6.317.3.4 CopySortedTo()	938
6.317.3.5 Get()	938
6.317.3.6 GetInsertTime()	938
6.317.3.7 GetLastUsedInputHeader()	939
6.317.3.8 Remove()	939
6.317.4 Property Documentation	939
6.317.4.1 Count	939
6.317.4.2 Full	940
6.318 SimulationInputHeader Struct Reference	940
6.318.1 Detailed Description	940
6.318.2 Member Data Documentation	940
6.318.2.1 InterpAlpha	940
6.318.2.2 InterpFrom	941
6.318.2.3 InterpTo	941
6.318.2.4 SIZE	941
6.318.2.5 Tick	941
6.318.2.6 WORD_COUNT	941
6.319 SimulationMessage Struct Reference	941
6.319.1 Detailed Description	944
6.319.2 Member Function Documentation	944
6.319.2.1 Allocate()	944
6.319.2.2 CanAllocateUserPayload()	944

6.319.2.3 Clone()	944
6.319.2.4 GetData()	945
6.319.2.5 GetFlag()	945
6.319.2.6 IsTargeted()	945
6.319.2.7 ReadInt()	946
6.319.2.8 ReadNetworkedObjectRef()	946
6.319.2.9 ReadVector3()	946
6.319.2.10 ReferenceCountAdd()	947
6.319.2.11 ReferenceCountSub()	947
6.319.2.12 SetDummy()	947
6.319.2.13 SetNotTickAligned()	947
6.319.2.14 SetStatic()	947
6.319.2.15 SetTarget()	947
6.319.2.16 SetUnreliable()	948
6.319.2.17 ToString() [1/2]	948
6.319.2.18 ToString() [2/2]	948
6.319.2.19 WriteInt()	948
6.319.2.20 WriteNetworkedObjectRef()	949
6.319.2.21 WriteVector3()	949
6.319.3 Member Data Documentation	949
6.319.3.1 Capacity	949
6.319.3.2 FLAG_DUMMY	949
6.319.3.3 FLAG_INTERNAL	950
6.319.3.4 FLAG_NOT_TICK_ALIGNED	950
6.319.3.5 FLAG_REMOTE	950
6.319.3.6 FLAG_STATIC	950
6.319.3.7 FLAG_TARGET_PLAYER	950
6.319.3.8 FLAG_TARGET_SERVER	950
6.319.3.9 FLAG_UNRELIABLE	951
6.319.3.10 FLAG_USER_FLAGS_START	951
6.319.3.11 FLAG_USER_MESSAGE	951
6.319.3.12 Flags	951
6.319.3.13 FLAGS_RESERVED	951
6.319.3.14 FLAGS_RESERVED_BITS	951
6.319.3.15 MAX_PAYLOAD_SIZE	952
6.319.3.16 Offset	952
6.319.3.17 References	952
6.319.3.18 SIZE	952
6.319.3.19 Source	952
6.319.3.20 Target	952
6.319.3.21 Tick	953
6.319.4 Property Documentation	953

6.319.4.1 IsUnreliable	953
6.320 SimulationMessagePtr Struct Reference	953
6.320.1 Detailed Description	953
6.320.2 Member Data Documentation	953
6.320.2.1 Message	953
6.321 SimulationRuntimeConfig Struct Reference	953
6.321.1 Detailed Description	954
6.321.2 Member Data Documentation	954
6.321.2.1 HostPlayer	954
6.321.2.2 MasterClient	954
6.321.2.3 PlayerMaxCount	954
6.321.2.4 ServerMode	955
6.321.2.5 TickRate	955
6.321.2.6 Topology	955
6.322 INetBitWriteStream Interface Reference	955
6.322.1 Detailed Description	956
6.322.2 Member Function Documentation	956
6.322.2.1 WriteBoolean()	956
6.322.2.2 WriteBytesAligned()	956
6.322.2.3 WriteInt32()	956
6.322.2.4 WriteInt32VarLength() [1/2]	957
6.322.2.5 WriteInt32VarLength() [2/2]	957
6.322.2.6 WriteUInt64VarLength()	957
6.322.3 Property Documentation	958
6.322.3.1 OffsetBits	958
6.323 INetPeerGroupCallbacks Interface Reference	958
6.323.1 Detailed Description	959
6.323.2 Member Function Documentation	959
6.323.2.1 OnConnected()	959
6.323.2.2 OnConnectionAttempt()	959
6.323.2.3 OnConnectionFailed()	959
6.323.2.4 OnConnectionRequest()	960
6.323.2.5 OnDisconnected()	960
6.323.2.6 OnNotifyData()	960
6.323.2.7 OnNotifyDelivered()	962
6.323.2.8 OnNotifyDispose()	962
6.323.2.9 OnNotifyLost()	962
6.323.2.10 OnReliableData()	963
6.323.2.11 OnUnconnectedData()	963
6.323.2.12 OnUnreliableData()	963
6.324 INetSocket Interface Reference	964
6.324.1 Detailed Description	964

6.324.2 Member Function Documentation	964
6.324.2.1 Bind()	964
6.324.2.2 CanFragment()	965
6.324.2.3 Create()	965
6.324.2.4 DeleteEncryptionKey()	965
6.324.2.5 Destroy()	966
6.324.2.6 Initialize()	966
6.324.2.7 Poll()	966
6.324.2.8 Receive()	967
6.324.2.9 Send()	967
6.324.2.10 SetupEncryption()	968
6.324.3 Property Documentation	968
6.324.3.1 SupportsMultiThreading	968
6.325 NetAddress Struct Reference	968
6.325.1 Detailed Description	969
6.325.2 Member Function Documentation	969
6.325.2.1 Any()	969
6.325.2.2 AnyIPv6()	970
6.325.2.3 CreateFromIpPort()	970
6.325.2.4 Equals() [1/2]	971
6.325.2.5 Equals() [2/2]	971
6.325.2.6 FromActorId()	971
6.325.2.7 GetHashCode()	971
6.325.2.8 LocalhostIPv4()	971
6.325.2.9 LocalhostIPv6()	972
6.325.2.10 ToString()	972
6.325.3 Property Documentation	972
6.325.3.1 ActorId	972
6.325.3.2 HasAddress	973
6.325.3.3 IsIPv4	973
6.325.3.4 IsIPv6	973
6.325.3.5 IsRelayAddr	973
6.325.3.6 IsValid	973
6.326 NetAddress.EqualityComparer Class Reference	973
6.326.1 Detailed Description	974
6.326.2 Member Function Documentation	974
6.326.2.1 Equals()	974
6.326.2.2 GetHashCode()	974
6.327 NetBitBuffer Struct Reference	974
6.327.1 Detailed Description	978
6.327.2 Member Function Documentation	978
6.327.2.1 Allocate()	978

6.327.2.2 CanRead()	979
6.327.2.3 CanWrite()	979
6.327.2.4 CheckBitCount()	979
6.327.2.5 Clear()	980
6.327.2.6 GetDataPointer()	980
6.327.2.7 GetOffset()	980
6.327.2.8 PadToByteBoundary()	981
6.327.2.9 PadToByteBoundaryAndGetPtr()	981
6.327.2.10 PeekBoolean()	981
6.327.2.11 ReadBoolean()	981
6.327.2.12 ReadByte()	981
6.327.2.13 ReadBytesAligned() [1/2]	982
6.327.2.14 ReadBytesAligned() [2/2]	982
6.327.2.15 ReadDouble()	982
6.327.2.16 ReadInt16()	983
6.327.2.17 ReadInt32()	983
6.327.2.18 ReadInt32VarLength() [1/2]	983
6.327.2.19 ReadInt32VarLength() [2/2]	983
6.327.2.20 ReadInt64()	984
6.327.2.21 ReadInt64VarLength()	984
6.327.2.22 ReadSingle()	984
6.327.2.23 ReadString() [1/2]	985
6.327.2.24 ReadString() [2/2]	985
6.327.2.25 ReadUInt16()	985
6.327.2.26 ReadUInt32()	986
6.327.2.27 ReadUInt32VarLength() [1/2]	986
6.327.2.28 ReadUInt32VarLength() [2/2]	986
6.327.2.29 ReadUInt64()	987
6.327.2.30 ReadUInt64VarLength()	987
6.327.2.31 Release()	987
6.327.2.32 ReleaseRef()	988
6.327.2.33 ReplaceDataFromBlockWithTemp()	988
6.327.2.34 SeekToByteBoundary()	988
6.327.2.35 SetBufferLengthBytes()	988
6.327.2.36 Write()	989
6.327.2.37 WriteBoolean()	989
6.327.2.38 WriteByte()	989
6.327.2.39 WriteBytesAligned() [1/2]	990
6.327.2.40 WriteBytesAligned() [2/2]	990
6.327.2.41 WriteDouble()	990
6.327.2.42 WriteInt16()	991
6.327.2.43 WriteInt32()	991

6.327.2.44 WriteInt32AtOffset()	991
6.327.2.45 WriteInt32VarLength() [1/2]	992
6.327.2.46 WriteInt32VarLength() [2/2]	992
6.327.2.47 WriteInt64()	992
6.327.2.48 WriteInt64VarLength()	993
6.327.2.49 WriteSingle()	993
6.327.2.50 WriteSlow()	993
6.327.2.51 WriteString() [1/2]	994
6.327.2.52 WriteString() [2/2]	994
6.327.2.53 WriteUInt16()	994
6.327.2.54 WriteUInt32()	994
6.327.2.55 WriteUInt32VarLength() [1/2]	995
6.327.2.56 WriteUInt32VarLength() [2/2]	995
6.327.2.57 WriteUInt64()	995
6.327.2.58 WriteUInt64AtOffset()	996
6.327.2.59 WriteUInt64VarLength()	996
6.327.3 Member Data Documentation	996
6.327.3.1 Address	996
6.327.4 Property Documentation	997
6.327.4.1 BytesRemaining	997
6.327.4.2 Data	997
6.327.4.3 Done	997
6.327.4.4 DoneOrOverflow	997
6.327.4.5 IsOnEvenByte	997
6.327.4.6 LengthBits	998
6.327.4.7 LengthBytes	998
6.327.4.8 MoreToRead	998
6.327.4.9 OffsetBits	998
6.327.4.10 OffsetBitsUnsafe	998
6.327.4.11 OffsetBytes	998
6.327.4.12 Overflow	999
6.328 NetBitBuffer.Offset Struct Reference	999
6.328.1 Detailed Description	999
6.328.2 Constructor & Destructor Documentation	999
6.328.2.1 Offset()	999
6.328.3 Member Function Documentation	999
6.328.3.1 GetLength()	1000
6.329 NetBitBufferList Struct Reference	1000
6.329.1 Detailed Description	1000
6.329.2 Member Function Documentation	1000
6.329.2.1 AddFirst()	1000
6.329.2.2 AddLast()	1001

6.329.2.3 IsInList()	1001
6.329.2.4 Remove()	1001
6.329.2.5 RemoveHead()	1002
6.330 NetBitBufferNull Struct Reference	1002
6.330.1 Detailed Description	1003
6.330.2 Member Function Documentation	1003
6.330.2.1 PadToByteBoundary()	1003
6.330.2.2 WriteBoolean()	1003
6.330.2.3 WriteByte()	1003
6.330.2.4 WriteBytesAligned()	1004
6.330.2.5 WriteInt32()	1004
6.330.2.6 WriteInt32VarLength() [1/2]	1004
6.330.2.7 WriteInt32VarLength() [2/2]	1005
6.330.2.8 WriteUInt32VarLength() [1/2]	1005
6.330.2.9 WriteUInt32VarLength() [2/2]	1005
6.330.2.10 WriteUInt64VarLength()	1006
6.330.3 Property Documentation	1006
6.330.3.1 OffsetBits	1006
6.331 NetBitBufferSerializer Struct Reference	1006
6.331.1 Detailed Description	1007
6.331.2 Member Function Documentation	1007
6.331.2.1 Check()	1007
6.331.2.2 Reader()	1008
6.331.2.3 Serialize() [1/7]	1008
6.331.2.4 Serialize() [2/7]	1008
6.331.2.5 Serialize() [3/7]	1009
6.331.2.6 Serialize() [4/7]	1009
6.331.2.7 Serialize() [5/7]	1009
6.331.2.8 Serialize() [6/7]	1010
6.331.2.9 Serialize() [7/7]	1010
6.331.2.10 Writer()	1010
6.331.3 Property Documentation	1010
6.331.3.1 Buffer	1011
6.331.3.2 Reading	1011
6.331.3.3 Writing	1011
6.332 NetCommandAccepted Struct Reference	1011
6.332.1 Detailed Description	1011
6.333 NetCommandConnect Struct Reference	1011
6.333.1 Detailed Description	1012
6.334 NetCommandDisconnect Struct Reference	1012
6.334.1 Detailed Description	1013
6.335 NetCommandHeader Struct Reference	1013

6.335.1 Detailed Description	1013
6.335.2 Member Function Documentation	1013
6.335.2.1 Create()	1013
6.336 NetCommandRefused Struct Reference	1014
6.336.1 Detailed Description	1014
6.337 NetConfig Struct Reference	1014
6.337.1 Detailed Description	1015
6.337.2 Member Data Documentation	1016
6.337.2.1 Address	1016
6.337.2.2 ConnectAttempts	1016
6.337.2.3 ConnectInterval	1016
6.337.2.4 ConnectionDefaultRtt	1016
6.337.2.5 ConnectionGroups	1016
6.337.2.6 ConnectionPingInterval	1017
6.337.2.7 ConnectionSendBuffers	1017
6.337.2.8 ConnectionShutdownTime	1017
6.337.2.9 ConnectionTimeout	1017
6.337.2.10 MaxConnections	1017
6.337.2.11 Notify	1017
6.337.2.12 OperationExpireTime	1018
6.337.2.13 PacketSize	1018
6.337.2.14 Simulation	1018
6.337.2.15 SocketRecvBuffer	1018
6.337.2.16 SocketSendBuffer	1018
6.337.3 Property Documentation	1018
6.337.3.1 ConnectionsPerGroup	1018
6.337.3.2 Defaults	1019
6.337.3.3 PacketSizeInBits	1019
6.338 NetConfigNotify Struct Reference	1019
6.338.1 Detailed Description	1019
6.338.2 Member Data Documentation	1020
6.338.2.1 AckForceCount	1020
6.338.2.2 AckForceTimeout	1020
6.338.2.3 AckMaskBytes	1020
6.338.2.4 SequenceBytes	1020
6.338.2.5 WindowSize	1020
6.338.3 Property Documentation	1020
6.338.3.1 AckMaskBits	1021
6.338.3.2 Defaults	1021
6.338.3.3 SequenceBounds	1021
6.339 NetConfigSimulation Struct Reference	1021
6.339.1 Detailed Description	1022

6.339.2 Member Function Documentation	1022
6.339.2.1 WithLossNotifySequences()	1022
6.339.3 Member Data Documentation	1022
6.339.3.1 DelayOscillator	1022
6.339.3.2 DuplicateChance	1022
6.339.3.3 LossNotifySequences	1023
6.339.3.4 LossNotifySequencesLength	1023
6.339.3.5 LossOscillator	1023
6.339.4 Property Documentation	1023
6.339.4.1 Defaults	1023
6.340 NetConfigSimulationOscillator Struct Reference	1023
6.340.1 Detailed Description	1024
6.340.2 Member Enumeration Documentation	1024
6.340.2.1 WaveShape	1024
6.340.3 Member Function Documentation	1024
6.340.3.1 GetCurveValue()	1024
6.340.4 Member Data Documentation	1025
6.340.4.1 Additional	1025
6.340.4.2 Max	1025
6.340.4.3 Min	1025
6.340.4.4 Period	1025
6.340.4.5 Shape	1026
6.340.4.6 Threshold	1026
6.341 NetConnection Struct Reference	1026
6.341.1 Detailed Description	1026
6.341.2 Member Function Documentation	1026
6.341.2.1 ToString()	1027
6.341.3 Property Documentation	1027
6.341.3.1 Active	1027
6.341.3.2 ConnectionStatus	1027
6.341.3.3 LocalConnectionId	1027
6.341.3.4 RemoteAddress	1027
6.341.3.5 RemoteConnectionId	1027
6.341.3.6 RoundTripTime	1028
6.342 NetConnectionId Struct Reference	1028
6.342.1 Detailed Description	1028
6.342.2 Member Function Documentation	1029
6.342.2.1 Equals() [1/2]	1029
6.342.2.2 Equals() [2/2]	1029
6.342.2.3 GetHashCode()	1029
6.342.2.4 operator"!="()	1029
6.342.2.5 operator=="()	1029

6.342.3 Member Data Documentation	1030
6.342.3.1 Group	1030
6.342.3.2 GroupIndex	1030
6.343 NetConnectionId.EqualityComparer Class Reference	1030
6.343.1 Detailed Description	1030
6.343.2 Member Function Documentation	1030
6.343.2.1 Equals()	1030
6.343.2.2 GetHashCode()	1031
6.344 NetConnectionMap Struct Reference	1031
6.344.1 Detailed Description	1032
6.344.2 Member Enumeration Documentation	1032
6.344.2.1 EntryState	1032
6.344.3 Member Function Documentation	1032
6.344.3.1 Allocate()	1033
6.344.3.2 Dispose()	1033
6.344.3.3 Find() [1/2]	1033
6.344.3.4 Find() [2/2]	1034
6.344.3.5 FindByIndex()	1034
6.344.3.6 Insert()	1034
6.344.3.7 Remap()	1035
6.344.3.8 Remove()	1035
6.344.3.9 TryFindByIndex()	1036
6.344.4 Property Documentation	1036
6.344.4.1 ConnectionsBuffer	1036
6.344.4.2 Count	1036
6.344.4.3 CountUsed	1036
6.344.4.4 Full	1037
6.345 NetConnectionMap.Iterator Struct Reference	1037
6.345.1 Detailed Description	1037
6.345.2 Constructor & Destructor Documentation	1037
6.345.2.1 Iterator()	1037
6.345.3 Member Function Documentation	1038
6.345.3.1 Next()	1038
6.345.4 Property Documentation	1038
6.345.4.1 Current	1038
6.345.4.2 IsValid	1038
6.346 NetPeer Struct Reference	1038
6.346.1 Detailed Description	1040
6.346.2 Member Function Documentation	1040
6.346.2.1 Destroy()	1040
6.346.2.2 GetConfigPointer()	1041
6.346.2.3 GetGroup()	1041

6.346.2.4 Initialize() [1/2]	1041
6.346.2.5 Initialize() [2/2]	1042
6.346.2.6 Recv() [1/2]	1042
6.346.2.7 Recv() [2/2]	1043
6.346.2.8 RemapAddress()	1043
6.346.2.9 Send() [1/2]	1043
6.346.2.10 Send() [2/2]	1044
6.346.2.11 Update() [1/2]	1044
6.346.2.12 Update() [2/2]	1044
6.346.3 Member Data Documentation	1045
6.346.3.1 DEFAULT_HEADERS	1045
6.346.3.2 MAX_MTU_BITS_PAYLOAD	1045
6.346.3.3 MAX_MTU_BYTES_PAYLOAD	1045
6.346.3.4 MAX_MTU_BYTES_TOTAL	1045
6.346.3.5 MAX_PACKET_BYTES_PAYLOAD	1045
6.346.3.6 MAX_PACKET_BYTES_TOTAL	1046
6.346.4 Property Documentation	1046
6.346.4.1 Address	1046
6.346.4.2 Config	1046
6.346.4.3 GroupCount	1046
6.346.4.4 IsShutdown	1046
6.347 NetPeerGroup Struct Reference	1046
6.347.1 Detailed Description	1049
6.347.2 Member Function Documentation	1049
6.347.2.1 ChangeConnectionAddressDuringConnecting()	1049
6.347.2.2 Connect() [1/2]	1049
6.347.2.3 Connect() [2/2]	1049
6.347.2.4 ConnectionIterator()	1050
6.347.2.5 Disconnect()	1050
6.347.2.6 GetConnection()	1051
6.347.2.7 GetConnectionByIndex()	1051
6.347.2.8 GetConnectionIdleTime()	1051
6.347.2.9 GetNotifyDataBuffer()	1052
6.347.2.10 GetUnreliableDataBuffer()	1052
6.347.2.11 SendNotifyDataBuffer()	1053
6.347.2.12 SendReliable()	1053
6.347.2.13 SendUnconnectedData()	1054
6.347.2.14 SendUnreliableDataBuffer()	1054
6.347.2.15 TryGetConnectionByIndex()	1054
6.347.2.16 Update()	1055
6.347.3 Property Documentation	1055
6.347.3.1 ConnectionCount	1055

6.347.3.2 Group	1055
6.347.3.3 Time	1056
6.348 NetSendEnvelope Struct Reference	1056
6.348.1 Detailed Description	1056
6.348.2 Member Data Documentation	1056
6.348.2.1 SendTime	1056
6.348.2.2 Sequence	1056
6.348.2.3 UserData	1057
6.349 NetSocket Struct Reference	1057
6.349.1 Detailed Description	1057
6.349.2 Member Data Documentation	1057
6.349.2.1 Handle	1057
6.349.2.2 NativeSocket	1057
6.349.3 Property Documentation	1058
6.349.3.1 IsCreated	1058
6.350 ReliableHeader Struct Reference	1058
6.350.1 Detailed Description	1058
6.350.2 Member Function Documentation	1058
6.350.2.1 GetData()	1058
6.350.3 Member Data Documentation	1059
6.350.3.1 Id	1059
6.350.3.2 Next	1059
6.350.3.3 Prev	1059
6.350.3.4 SIZE	1059
6.351 ReliableId Struct Reference	1059
6.351.1 Detailed Description	1060
6.351.2 Member Data Documentation	1060
6.351.2.1 _padding	1060
6.351.2.2 Key	1061
6.351.2.3 Sequence	1061
6.351.2.4 SIZE	1061
6.351.2.5 SliceLength	1061
6.351.2.6 Source	1061
6.351.2.7 SourceSend	1061
6.351.2.8 Target	1062
6.351.2.9 TotalLength	1062
6.351.3 Property Documentation	1062
6.351.3.1 SourceCombined	1062
6.352 ReliableKey Struct Reference	1062
6.352.1 Detailed Description	1063
6.352.2 Member Function Documentation	1063
6.352.2.1 FromInts()	1063

6.352.2.2 FromULongs()	1063
6.352.2.3 GetInts()	1064
6.352.2.4 GetULongs()	1064
6.352.3 Member Data Documentation	1064
6.352.3.1 Data	1064
6.352.3.2 SIZE	1065
6.353 ReliableList Struct Reference	1065
6.353.1 Detailed Description	1065
6.353.2 Member Function Documentation	1065
6.353.2.1 AddAfter()	1065
6.353.2.2 AddBefore()	1066
6.353.2.3 AddFirst()	1066
6.353.2.4 AddLast()	1066
6.353.2.5 Remove()	1067
6.353.2.6 RemoveHead()	1067
6.353.3 Member Data Documentation	1067
6.353.3.1 Count	1067
6.353.3.2 Head	1067
6.353.3.3 Tail	1067
6.354 StunServers.StunServer Class Reference	1068
6.354.1 Detailed Description	1068
6.355 StartGameArgs Struct Reference	1068
6.355.1 Detailed Description	1070
6.355.2 Member Function Documentation	1070
6.355.2.1 ToString()	1070
6.355.3 Member Data Documentation	1070
6.355.3.1 Address	1070
6.355.3.2 AuthValues	1071
6.355.3.3 Config	1071
6.355.3.4 ConnectionToken	1071
6.355.3.5 CustomCallbackInterfaces	1071
6.355.3.6 CustomLobbyName	1071
6.355.3.7 CustomPhotonAppSettings	1071
6.355.3.8 CustomPublicAddress	1072
6.355.3.9 CustomSTUNServer	1072
6.355.3.10 DisableNATPunchthrough	1072
6.355.3.11 EnableClientSessionCreation	1072
6.355.3.12 GameMode	1072
6.355.3.13 HostMigrationResume	1072
6.355.3.14 HostMigrationToken	1073
6.355.3.15 IsOpen	1073
6.355.3.16 IsVisible	1073

6.355.3.17 MatchmakingMode	1073
6.355.3.18 ObjectInitializer	1073
6.355.3.19 ObjectProvider	1073
6.355.3.20 OnGameStarted	1074
6.355.3.21 PlayerCount	1074
6.355.3.22 Scene	1074
6.355.3.23 SceneManager	1074
6.355.3.24 SessionName	1074
6.355.3.25 SessionNameGenerator	1074
6.355.3.26 SessionProperties	1075
6.355.3.27 StartGameCancellationToken	1075
6.355.3.28 Updater	1075
6.355.3.29 UseCachedRegions	1075
6.355.3.30 UseDefaultPhotonCloudPorts	1075
6.356 StartGameResult Class Reference	1075
6.356.1 Detailed Description	1076
6.356.2 Member Function Documentation	1076
6.356.2.1 ToString()	1076
6.356.3 Property Documentation	1076
6.356.3.1 ErrorMessage	1076
6.356.3.2 Ok	1077
6.356.3.3 ShutdownReason	1077
6.356.3.4 StackTrace	1077
6.357 BehaviourStatisticsManager Class Reference	1077
6.357.1 Detailed Description	1077
6.357.2 Property Documentation	1077
6.357.2.1 CompletedSnapshot	1077
6.358 BehaviourStatisticsSnapshot Class Reference	1078
6.358.1 Detailed Description	1078
6.358.2 Property Documentation	1078
6.358.2.1 FixedUpdateNetworkExecutionCount	1078
6.358.2.2 FixedUpdateNetworkExecutionTime	1078
6.358.2.3 RenderExecutionCount	1078
6.358.2.4 RenderExecutionTime	1079
6.359 FusionStatisticsManager Class Reference	1079
6.359.1 Detailed Description	1079
6.359.2 Property Documentation	1079
6.359.2.1 CompleteSnapshot	1079
6.359.2.2 ObjectStatisticsManager	1079
6.360 FusionStatisticsSnapshot Class Reference	1079
6.360.1 Detailed Description	1081
6.360.2 Property Documentation	1081

6.360.2.1 ForwardTicks	1081
6.360.2.2 GeneralAllocMemoryFreeInBytes	1081
6.360.2.3 GeneralAllocMemoryUsedInBytes	1081
6.360.2.4 InBandwidth	1081
6.360.2.5 InObjectUpdates	1081
6.360.2.6 InPackets	1082
6.360.2.7 InputInBandwidth	1082
6.360.2.8 InputOutBandwidth	1082
6.360.2.9 InputReceiveDelta	1082
6.360.2.10 InterpolationOffset	1082
6.360.2.11 InterpolationSpeed	1082
6.360.2.12 ObjectsAllocMemoryFreeInBytes	1083
6.360.2.13 ObjectsAllocMemoryUsedInBytes	1083
6.360.2.14 OutBandwidth	1083
6.360.2.15 OutObjectUpdates	1083
6.360.2.16 OutPackets	1083
6.360.2.17 Resimulations	1083
6.360.2.18 RoundTripTime	1084
6.360.2.19 SimulationSpeed	1084
6.360.2.20 SimulationTimeOffset	1084
6.360.2.21 StateReceiveDelta	1084
6.360.2.22 TimeResets	1084
6.360.2.23 WordsReadCount	1084
6.360.2.24 WordsReadSize	1085
6.360.2.25 WordsWrittenCount	1085
6.360.2.26 WordsWrittenSize	1085
6.361 LagCompensationStatisticsSnapshot Class Reference	1085
6.361.1 Detailed Description	1086
6.361.2 Property Documentation	1086
6.361.2.1 AddOnBufferTime	1086
6.361.2.2 AddOnBVHTime	1086
6.361.2.3 AdvanceBufferTime	1086
6.361.2.4 BVHMaxDeep	1086
6.361.2.5 BVHNodesCount	1086
6.361.2.6 HitboxesCount	1087
6.361.2.7 RefitBVHTime	1087
6.361.2.8 TotalElapsedTime	1087
6.361.2.9 UpdateBufferTime	1087
6.361.2.10 UpdateBVHTime	1087
6.362 MemoryStatisticsSnapshot Struct Reference	1087
6.362.1 Detailed Description	1088
6.362.2 Member Enumeration Documentation	1088

6.362.2.1 TargetAllocator	1088
6.362.3 Member Data Documentation	1088
6.362.3.1 BUCKET_COUNT	1088
6.362.3.2 BucketFreeBlocksCount	1089
6.362.3.3 BucketFullBlocksCount	1089
6.362.3.4 BucketUsedBlocksCount	1089
6.362.3.5 TotalFreeBlocks	1089
6.363 NetworkObjectStatisticsManager Class Reference	1089
6.363.1 Detailed Description	1089
6.363.2 Member Function Documentation	1090
6.363.2.1 ClearMonitoredNetworkObjects()	1090
6.363.2.2 GetNetworkObjectStatistics()	1090
6.363.2.3 MonitorNetworkObjectStatistics()	1090
6.364 NetworkObjectStatisticsSnapshot Class Reference	1090
6.364.1 Detailed Description	1091
6.364.2 Property Documentation	1091
6.364.2.1 InBandwidth	1091
6.364.2.2 InPackets	1091
6.364.2.3 OutBandwidth	1091
6.364.2.4 OutPackets	1091
6.365 Tick Struct Reference	1092
6.365.1 Detailed Description	1093
6.365.2 Member Function Documentation	1093
6.365.2.1 CompareTo()	1093
6.365.2.2 Equals() [1/2]	1093
6.365.2.3 Equals() [2/2]	1094
6.365.2.4 GetHashCode()	1094
6.365.2.5 Next()	1094
6.365.2.6 operator bool()	1095
6.365.2.7 operator int()	1095
6.365.2.8 operator Tick()	1095
6.365.2.9 operator"!=()	1097
6.365.2.10 operator<()	1097
6.365.2.11 operator<=()	1097
6.365.2.12 operator==(())	1097
6.365.2.13 operator>()	1098
6.365.2.14 operator>=()	1098
6.365.2.15 ToString()	1098
6.365.3 Member Data Documentation	1098
6.365.3.1 ALIGNMENT	1098
6.365.3.2 Raw	1098
6.365.3.3 SIZE	1099

6.366 Tick.EqualityComparer Class Reference	1099
6.366.1 Detailed Description	1099
6.366.2 Member Function Documentation	1099
6.366.2.1 Equals()	1099
6.366.2.2 GetHashCode()	1100
6.367 Tick.RelationalComparer Class Reference	1100
6.367.1 Detailed Description	1100
6.367.2 Member Function Documentation	1100
6.367.2.1 Compare()	1100
6.368 TickAccumulator Struct Reference	1101
6.368.1 Detailed Description	1102
6.368.2 Member Function Documentation	1102
6.368.2.1 AddTicks()	1102
6.368.2.2 AddTime()	1102
6.368.2.3 Alpha()	1102
6.368.2.4 ConsumeTick()	1103
6.368.2.5 Start()	1103
6.368.2.6 StartNew()	1103
6.368.2.7 Stop()	1104
6.368.3 Property Documentation	1104
6.368.3.1 Pending	1104
6.368.3.2 Remainder	1104
6.368.3.3 Running	1104
6.368.3.4 TimeScale	1104
6.369 TickRate Struct Reference	1105
6.369.1 Detailed Description	1106
6.369.2 Member Enumeration Documentation	1106
6.369.2.1 ValidateResult	1106
6.369.3 Member Function Documentation	1107
6.369.3.1 ClampSelection()	1107
6.369.3.2 Get()	1107
6.369.3.3 GetDivisor()	1107
6.369.3.4 GetTickRate()	1108
6.369.3.5 Init()	1108
6.369.3.6 IsValid() [1/2]	1109
6.369.3.7 IsValid() [2/2]	1109
6.369.3.8 Resolve()	1109
6.369.3.9 ToArray()	1110
6.369.3.10 Validate()	1110
6.369.3.11 ValidateSelection()	1110
6.369.4 Property Documentation	1111
6.369.4.1 Available	1111

6.369.4.2 Client	1111
6.369.4.3 Count	1111
6.369.4.4 this[int index]	1111
6.370 TickRate.Resolved Struct Reference	1112
6.370.1 Detailed Description	1113
6.370.2 Member Data Documentation	1113
6.370.2.1 Client	1113
6.370.2.2 ClientSend	1113
6.370.2.3 Server	1113
6.370.2.4 ServerSend	1113
6.370.2.5 SIZE	1113
6.370.2.6 WORDS	1114
6.370.3 Property Documentation	1114
6.370.3.1 ClientSendDelta	1114
6.370.3.2 ClientTickDelta	1114
6.370.3.3 ClientTickStride	1114
6.370.3.4 ServerSendDelta	1114
6.370.3.5 ServerTickDelta	1114
6.370.3.6 ServerTickStride	1115
6.371 TickRate.Selection Struct Reference	1115
6.371.1 Detailed Description	1115
6.371.2 Member Data Documentation	1115
6.371.2.1 Client	1115
6.371.2.2 ClientSendIndex	1115
6.371.2.3 ServerIndex	1116
6.371.2.4 ServerSendIndex	1116
6.372 TickTimer Struct Reference	1116
6.372.1 Detailed Description	1117
6.372.2 Member Function Documentation	1117
6.372.2.1 CreateFromSeconds()	1117
6.372.2.2 CreateFromTicks()	1117
6.372.2.3 Expired()	1118
6.372.2.4 ExpiredOrNotRunning()	1118
6.372.2.5 RemainingTicks()	1119
6.372.2.6 RemainingTime()	1119
6.372.2.7 ToString()	1119
6.372.3 Property Documentation	1119
6.372.3.1 IsRunning	1120
6.372.3.2 None	1120
6.372.3.3 TargetTick	1120
6.373 Timer Struct Reference	1120
6.373.1 Detailed Description	1121

6.373.2 Member Function Documentation	1121
6.373.2.1 Reset()	1121
6.373.2.2 Restart()	1121
6.373.2.3 Start()	1121
6.373.2.4 StartNew()	1122
6.373.2.5 Stop()	1122
6.373.3 Property Documentation	1122
6.373.3.1 ElapsedInMilliseconds	1122
6.373.3.2 ElapsedInSeconds	1122
6.373.3.3 ElapsedInTicks	1122
6.373.3.4 IsRunning	1123
6.374 TimeSyncConfiguration Class Reference	1123
6.374.1 Detailed Description	1123
6.374.2 Member Data Documentation	1123
6.374.2.1 MaxLateInputs	1123
6.374.2.2 MaxLateSnapshots	1123
6.374.2.3 RedundantInputs	1124
6.374.2.4 RedundantSnapshots	1124
6.374.2.5 SampleWindowSeconds	1124
6.375 ToggleLeftAttribute Class Reference	1124
6.375.1 Detailed Description	1124
6.376 UnitAttribute Class Reference	1124
6.376.1 Detailed Description	1125
6.376.2 Constructor & Destructor Documentation	1125
6.376.2.1 UnitAttribute()	1125
6.376.3 Property Documentation	1125
6.376.3.1 Unit	1125
6.377 UnityAddressablesRuntimeKeyAttribute Class Reference	1125
6.377.1 Detailed Description	1126
6.378 UnityAssetGuidAttribute Class Reference	1126
6.378.1 Detailed Description	1126
6.379 UnityContextMenuAttribute Class Reference	1126
6.379.1 Detailed Description	1126
6.379.2 Constructor & Destructor Documentation	1126
6.379.2.1 UnityContextMenuAttribute()	1127
6.379.3 Property Documentation	1127
6.379.3.1 order	1127
6.380 UnityDelayedAttribute Class Reference	1127
6.380.1 Detailed Description	1127
6.380.2 Property Documentation	1127
6.380.2.1 order	1127
6.381 UnityFormerlySerializedAsAttribute Class Reference	1128

6.381.1 Detailed Description	1128
6.381.2 Constructor & Destructor Documentation	1128
6.381.2.1 UnityFormerlySerializedAsAttribute()	1128
6.382 UnityHeaderAttribute Class Reference	1128
6.382.1 Detailed Description	1129
6.382.2 Constructor & Destructor Documentation	1129
6.382.2.1 UnityHeaderAttribute()	1129
6.382.3 Property Documentation	1129
6.382.3.1 order	1129
6.383 UnityMinAttribute Class Reference	1129
6.383.1 Detailed Description	1129
6.383.2 Constructor & Destructor Documentation	1130
6.383.2.1 UnityMinAttribute()	1130
6.383.3 Property Documentation	1130
6.383.3.1 order	1130
6.384 UnityMultilineAttribute Class Reference	1130
6.384.1 Detailed Description	1130
6.384.2 Property Documentation	1130
6.384.2.1 order	1131
6.385 UnityNonReorderableAttribute Class Reference	1131
6.385.1 Detailed Description	1131
6.385.2 Property Documentation	1131
6.385.2.1 order	1131
6.386 UnityNonSerializedAttribute Class Reference	1131
6.386.1 Detailed Description	1131
6.387 UnityRangeAttribute Class Reference	1132
6.387.1 Detailed Description	1132
6.387.2 Constructor & Destructor Documentation	1132
6.387.2.1 UnityRangeAttribute()	1132
6.387.3 Property Documentation	1132
6.387.3.1 order	1132
6.388 UnityResourcePathAttribute Class Reference	1133
6.388.1 Detailed Description	1133
6.388.2 Constructor & Destructor Documentation	1133
6.388.2.1 UnityResourcePathAttribute()	1133
6.388.3 Property Documentation	1133
6.388.3.1 ResourceType	1133
6.389 UnitySerializeField Class Reference	1134
6.389.1 Detailed Description	1134
6.390 UnitySerializeReference Class Reference	1134
6.390.1 Detailed Description	1134
6.391 UnitySpaceAttribute Class Reference	1134

6.391.1 Detailed Description	1134
6.391.2 Constructor & Destructor Documentation	1135
6.391.2.1 UnitySpaceAttribute()	1135
6.391.3 Property Documentation	1135
6.391.3.1 order	1135
6.392 UnityTooltipAttribute Class Reference	1135
6.392.1 Detailed Description	1135
6.392.2 Constructor & Destructor Documentation	1135
6.392.2.1 UnityTooltipAttribute()	1136
6.392.3 Property Documentation	1136
6.392.3.1 order	1136
6.393 UTF32Tools Class Reference	1136
6.393.1 Detailed Description	1136
6.393.2 Member Function Documentation	1136
6.393.2.1 Convert() [1/2]	1136
6.393.2.2 Convert() [2/2]	1137
6.393.2.3 GetLength()	1137
6.394 UTF32Tools.CharEnumerator Struct Reference	1138
6.394.1 Detailed Description	1138
6.394.2 Member Function Documentation	1138
6.394.2.1 Dispose()	1138
6.394.2.2 MoveNext()	1139
6.394.2.3 Reset()	1139
6.394.3 Property Documentation	1139
6.394.3.1 Current	1139
6.395 UTF32Tools.ConversionResult Struct Reference	1139
6.395.1 Detailed Description	1140
6.395.2 Constructor & Destructor Documentation	1140
6.395.2.1 ConversionResult()	1140
6.395.3 Member Data Documentation	1140
6.395.3.1 CharacterCount	1140
6.395.3.2 CodePointCount	1140
6.396 Vector2Compressed Struct Reference	1140
6.396.1 Detailed Description	1141
6.396.2 Member Function Documentation	1141
6.396.2.1 Equals() [1/2]	1141
6.396.2.2 Equals() [2/2]	1142
6.396.2.3 GetHashCode()	1142
6.396.2.4 operator Vector2()	1142
6.396.2.5 operator Vector2Compressed()	1143
6.396.2.6 operator"!=()	1143
6.396.2.7 operator"==()	1144

6.396.3 Member Data Documentation	1144
6.396.3.1 xEncoded	1144
6.396.3.2 yEncoded	1144
6.396.4 Property Documentation	1144
6.396.4.1 X	1144
6.396.4.2 Y	1145
6.397 Vector3Compressed Struct Reference	1145
6.397.1 Detailed Description	1146
6.397.2 Member Function Documentation	1146
6.397.2.1 Equals() [1/2]	1146
6.397.2.2 Equals() [2/2]	1146
6.397.2.3 GetHashCode()	1147
6.397.2.4 operator Vector2()	1147
6.397.2.5 operator Vector3()	1147
6.397.2.6 operator Vector3Compressed() [1/2]	1148
6.397.2.7 operator Vector3Compressed() [2/2]	1148
6.397.2.8 operator"!=()	1148
6.397.2.9 operator==()	1149
6.397.3 Member Data Documentation	1149
6.397.3.1 xEncoded	1149
6.397.3.2 yEncoded	1149
6.397.3.3 zEncoded	1150
6.397.4 Property Documentation	1150
6.397.4.1 X	1150
6.397.4.2 Y	1150
6.397.4.3 Z	1150
6.398 Vector4Compressed Struct Reference	1150
6.398.1 Detailed Description	1151
6.398.2 Member Function Documentation	1151
6.398.2.1 Equals() [1/2]	1151
6.398.2.2 Equals() [2/2]	1152
6.398.2.3 GetHashCode()	1152
6.398.2.4 operator Vector4()	1152
6.398.2.5 operator Vector4Compressed()	1153
6.398.2.6 operator"!=()	1153
6.398.2.7 operator==()	1154
6.398.3 Member Data Documentation	1154
6.398.3.1 wEncoded	1154
6.398.3.2 xEncoded	1154
6.398.3.3 yEncoded	1154
6.398.3.4 zEncoded	1155
6.398.4 Property Documentation	1155

6.398.4.1 W	1155
6.398.4.2 X	1155
6.398.4.3 Y	1155
6.398.4.4 Z	1155
6.399 Versioning Class Reference	1155
6.399.1 Detailed Description	1156
6.399.2 Member Function Documentation	1156
6.399.2.1 AssemblyFileVersion()	1156
6.399.2.2 ShortVersion()	1156
6.399.3 Member Data Documentation	1157
6.399.3.1 InvalidVersion	1157
6.399.4 Property Documentation	1157
6.399.4.1 GetCurrentVersion	1157
6.400 WarnIfAttribute Class Reference	1157
6.400.1 Detailed Description	1158
6.400.2 Constructor & Destructor Documentation	1158
6.400.2.1 WarnIfAttribute()	1158
6.400.3 Member Data Documentation	1158
6.400.3.1 AsBox	1158
6.400.3.2 Message	1159
6.401 WeaverGeneratedAttribute Class Reference	1159
6.401.1 Detailed Description	1159
6.402 IExportedWordCount Interface Reference	1159
6.402.1 Detailed Description	1159
6.402.2 Property Documentation	1159
6.402.2.1 WordCount	1159
6.403 IPublicFacingInterface Interface Reference	1160
6.403.1 Detailed Description	1160
6.404 NetworkedWeavedStringAttribute Class Reference	1160
6.404.1 Detailed Description	1160
6.404.2 Constructor & Destructor Documentation	1160
6.404.2.1 NetworkedWeavedStringAttribute()	1160
6.404.3 Property Documentation	1161
6.404.3.1 CacheFieldName	1161
6.404.3.2 Capacity	1161

Chapter 1

Photon Fusion API Documentation

Welcome to the Photon [Fusion](#) API online documentation.

1.1 Main Fusion API

Class	Description
Fusion.NetworkRunner	Represents a Server or Client Simulation
Fusion.NetworkObject	This stores the object's network identity and manages the object's state and input authority
Fusion.NetworkProjectConfig	The core Fusion config file that is shared with all peers at startup
Fusion.NetworkBehaviour	Base class for Fusion network components, which are associated with a Fusion.NetworkObject
Fusion.NetworkTransform	Replicates a Unity Transform's position and rotation state

Chapter 2

Photon Fusion Overview

Fusion is a new high performance state synchronization networking library for Unity. **Fusion** is built with simplicity in mind to integrate naturally into the common Unity workflow, while also offering advanced features like data compression, client-side prediction and lag compensation out of the box.

Behind the covers, **Fusion** relies on a state-of-the-art compression algorithm to reduce bandwidth requirements with minimal CPU overhead. Data is transferred as partial chunks with eventual consistency. A fully configurable area-of-interest system is supplied to allow support for very high player counts.

The **Fusion** API is designed to be similar to regular Unity MonoBehaviour code. For example, RPCs and network state is defined with attributes on methods and properties of MonoBehaviour with no need for explicit serialization code and network objects can be defined as prefabs using all of Unity's most recent prefab features like nesting and variants.

Inputs, Networked Properties and RPCs provide the foundation for writing gameplay code with **Fusion**.

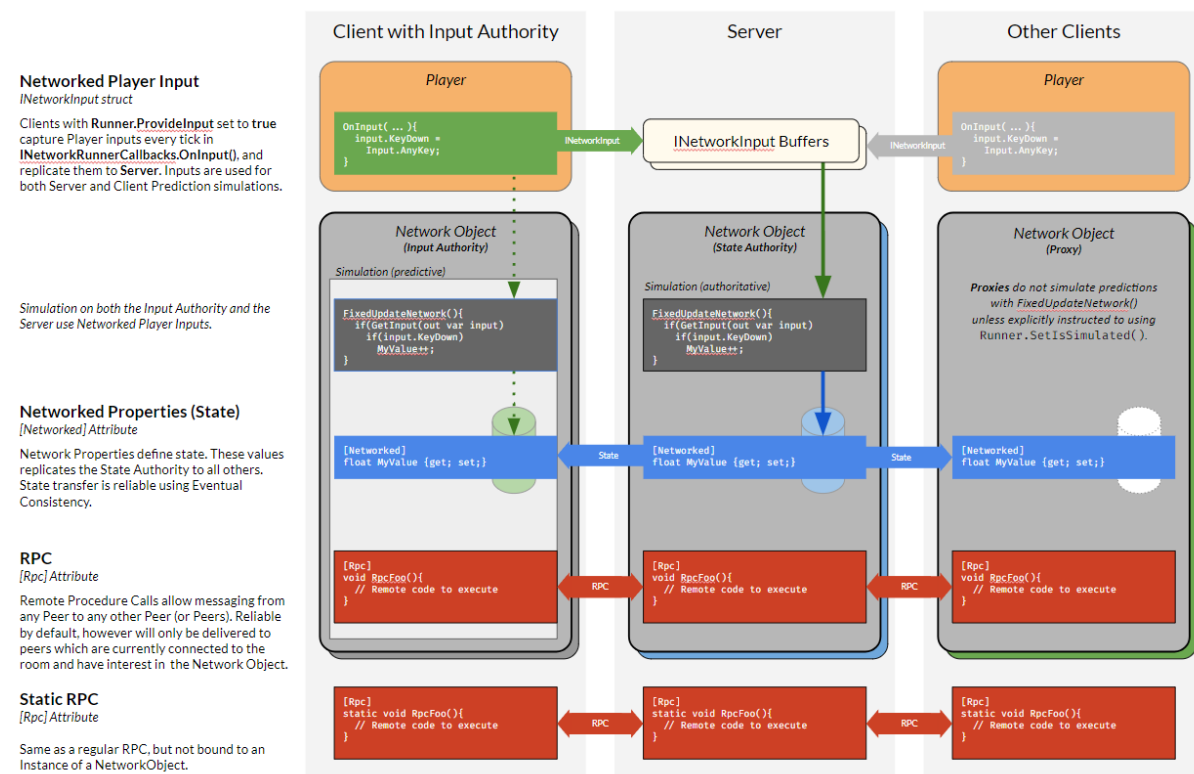


Figure 2.1 Overview of the Core Fusion APIs

Using a Networked Property in Fusion:

```
[Networked] public byte life { get; set; }
```

Using Network Input for client-side predicted movement:

```
public override void FixedUpdateNetwork
{
    if (GetInput(out NetworkInputData data))
    {
        data.direction.Normalize(); // normalize to prevent cheating with impossible inputs
        _characterController.Move(5 * data.direction * Runner.DeltaTime);
    }
}
```

Declaring a Remote Procedure Call (RPC) in Fusion:

```
[Rpc(RpcSources.InputAuthority, RpcTargets.StateAuthority)]
public void RPC_Configure(string name, Color color)
{
    playerName = name;
    playerColor = color;
}
```

2.1 Choosing the Right Mode

Fusion supports two fundamentally different network topologies with the same API as well as a single player mode with no network connection.

The first step when starting with Fusion is to choose between Server/Host and Shared mode.

The **Quadrant** provides a good starting point for deciding what mode is right for your application.



Figure 2.2 The Quadrant

2.2 Topology Differences

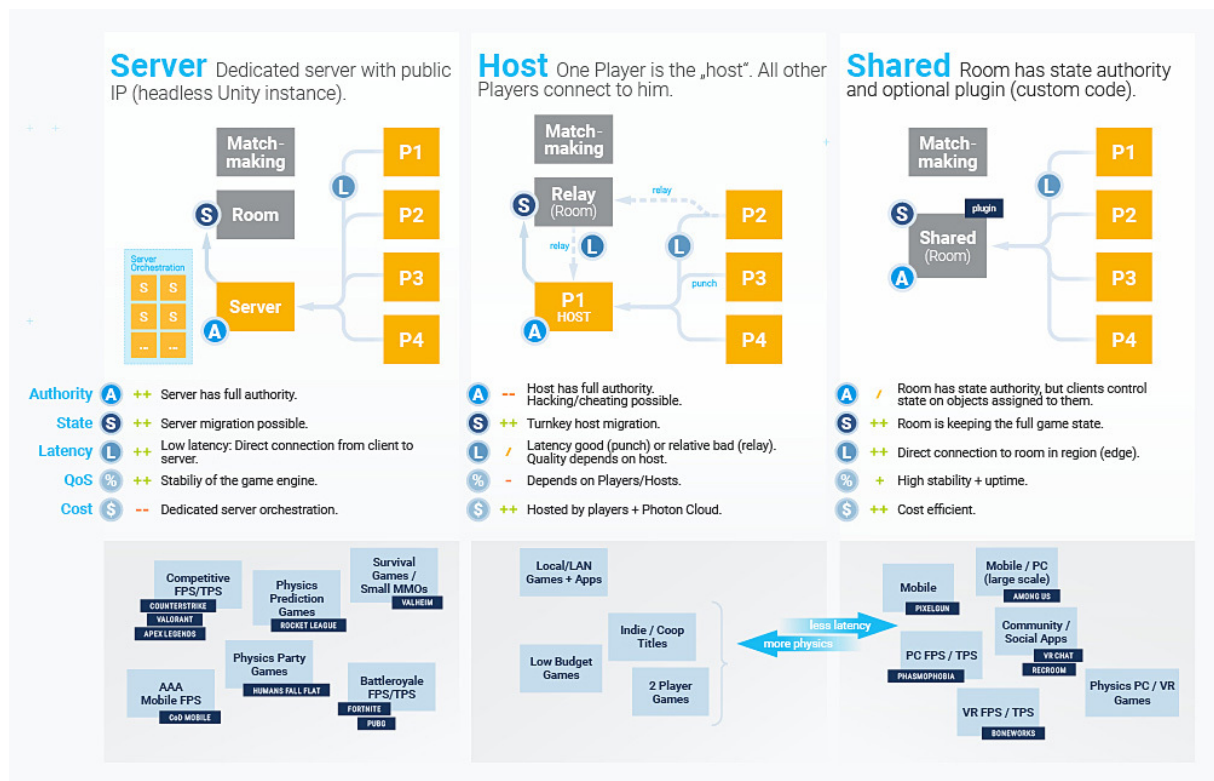


Figure 2.3 Fusion Network Topologies

2.2.1 Server Mode

In Server Mode the server has full and exclusive State Authority over all objects, no exceptions.

Clients can only modify networked objects by sending their input to the server (and have the server react to that input) or by requesting a change using an RPC.

The server application is built from the Unity project and runs a full headless Unity build. This headless build needs to be hosted on a server machine or a cloud hosted server. Photon does not provide servers for hosting a dedicated fusion server application.

2.2.1.1 Client Side Prediction

Client Side Prediction is a popular multiplayer architecture in which clients use their own inputs to predict their movement before receiving confirmation from the server. This allows the gameplay to feel snappy and hides latency.

In Fusion Server Mode, any changes a client makes directly to the networked state is only a local prediction, which will be overridden with actual authoritative snapshots from the server when those are received. This is known as reconciliation, as the client is rolled back to the server-provided state and re-simulated forward to the local (predicted) tick.

If previous predictions were accurate, this process is seamless. If not, the state will be updated and because the network state is separate from the rendering state, the rendering may either snap to this new state or use various forms of interpolation, error correction and smoothing to reduce the visual artifacts caused by the correction.

2.2.2 Host Mode

In Host Mode, the host acts as both a server and a client. The host has a local player and polls input for it and interpolates on rendering as expected of a client.

Overall the mode is equivalent to a dedicated server albeit much cheaper to run as no dedicated server hosting costs are incurred. This benefit comes in expense of losing a trustworthy authority; in other words a rogue host can cheat.

When running hosted mode from behind a firewall or a router, the Photon cloud transparently provides UDP punch through or package relay as needed,

Since the session is owned by the host, it will be lost if the host disconnects. [Fusion](#) does provide a host migration mechanism to allow transfer of network authority to a new client in the event that the current host is disconnected. Do note that, unlike Shared Mode, this requires special handling in client code.

2.2.3 Shared Mode

In shared mode, authority over network objects is distributed among all clients. Specifically, each client initially has State Authority over objects they spawn, but are free to release that State Authority to other clients. Optionally, clients may be allowed to take State Authority at will.

In shared mode features such as client side prediction and rollback are not available. Simulation always moves forward at the same tick rate on all clients.

The Shared Mode network session is owned by the Photon cloud and remains alive as long as any client is connected to it. The Photon cloud serves as a package relay and has full access to the network state with no need to run Unity, allowing for lightweight server logic and data validation (e.g. cheat protection) to be implemented without the need to spin up dedicated server hardware.

For those coming from Photon Unity Networking (PUN). Shared mode is in many ways similar to PUN, albeit more feature complete, faster, and with no run-time allocation overhead.

2.2.4 Cost

The same CCU costs apply for all modes. The server and clients all have to be connected to the Photon Cloud at all times for connection management. In Server Mode there are additional costs for hosting the dedicated server on a cloud service or your own hardware.

Chapter 3

Hierarchical Index

3.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Allocator	81
Allocator.Config	83
StaticConstructorAttribute	87
StaticFieldAttribute	88
StaticFieldResetMethodAttribute	89
AssetObject	102
TaskManager	102
AtomicInt	106
AuthorityMasks	109
Behaviour	110
Hitbox	189
NetworkEvents	489
NetworkObject	527
NetworkObjectPrefabData	566
NetworkRunner	631
SimulationBehaviour	925
HitboxManager	193
NetworkBehaviour	383
HitboxRoot	207
NetworkMecanimAnimator	524
NetworkTRSP	762
NetworkTransform	759
BinUtils	112
CapacityAttribute	117
CRC64	118
DynamicHeap	131
DynamicHeap.Ignore	135
DynamicHeap.Instance	135
EditorButtonAttribute	139
IDataEncryption	143
DataEncryptor	141
EncryptionConfig	146
EngineProfiler	146
FieldsMask< T >	160

FixedArray< T >	163
FixedArray< T >.Enumerator	170
FixedStorage	173
FloatUtils	177
FusionGlobalScriptableObjectAttribute	182
FusionGlobalScriptableObjectLoadResult	183
FusionGlobalScriptableObjectSourceAttribute	185
FusionMonoBehaviour	187
FusionScriptableObject	187
FusionGlobalScriptableObject< NetworkProjectConfigAsset >	179
NetworkProjectConfigAsset	620
FusionGlobalScriptableObject< T >	179
HeapConfiguration	187
HostMigrationConfig	213
HostMigrationToken	214
IAsyncOperation	220
ICoroutine	227
IElementReaderWriter< T >	228
IFixedStorage	230
_128	73
_16	74
_2	75
_256	75
_32	76
_4	77
_512	78
_64	79
_8	80
INetworkArray	233
NetworkArray< T >	372
INetworkAssetSource< T >	234
INetworkDictionary	236
NetworkDictionary< K, V >	473
INetworkInput	237
INetworkLinkedList	237
NetworkLinkedList< T >	507
INetworkObjectInitializer	238
NetworkObjectInitializerUnity	559
INetworkObjectProvider	238
NetworkObjectProviderDummy	567
INetworkRunnerUpdater	248
NetworkRunnerUpdaterDefault	699
INetworkSceneManager	249
INetworkStruct	253
Angle	90
FloatCompressed	174
NetworkBehaviourId	434
NetworkBool	456
NetworkButtons	459
NetworkId	495
NetworkObjectGuid	540
NetworkObjectHeader	550
NetworkObjectNestingKey	561
NetworkObjectTypeId	572
NetworkPhysicsInfo	582

NetworkPrefabId	585
NetworkPrefabRef	593
NetworkRNG	624
NetworkSceneInfo	711
NetworkString< TSize >	732
NetworkTRSPData	765
PlayerRef	771
Ptr	828
QuaternionCompressed	834
SceneRef	879
TickTimer	1116
Vector2Compressed	1140
Vector3Compressed	1145
Vector4Compressed	1150
_128	73
_16	74
_2	75
_256	75
_32	76
_4	77
_512	78
_64	79
_8	80
INetworkTRSPTeleport	254
NetworkTransform	759
IUnitySurrogate	255
IUnityValueSurrogate< T >	256
UnityValueSurrogate< T, TReaderWriter >	265
UnitySurrogateBase	263
UnityArraySurrogate< T, ReaderWriter >	257
UnityDictionarySurrogate< TKeyType, TKeyReaderWriter, TValueType, TValueReaderWriter >	259
UnityLinkedListSurrogate< T, ReaderWriter >	261
UnityValueSurrogate< T, TReaderWriter >	265
InterpolatedErrorCorrectionSettings	267
LagCompensatedHit	276
AABB	279
BoxOverlapQueryParams	284
BVHDraw	286
BVHNodeDrawInfo	286
ColliderDrawInfo	287
HitboxColliderContainerDraw	289
LagCompensatedExt	290
LagCompensationDraw	291
LagCompensationUtils.ContactData	292
PositionRotationQueryParams	293
Query	294
BoxOverlapQuery	281
RaycastQuery	301
RaycastAllQuery	300
SphereOverlapQuery	305
QueryParams	298
RaycastQueryParams	303
SnapshotHistoryDraw	305
SphereOverlapQueryParams	307
LagCompensationSettings	309
LobbyInfo	311
Mask256	312

Maths	317
Maths.FastAbs	334
Native	336
NestedComponentUtilities	364
NetworkArray< T >.Enumerator	379
NetworkArrayExtensions	381
NetworkArrayReadOnly< T >	382
NetworkAssemblyIgnoreAttribute	383
NetworkAssemblyWeavedAttribute	383
NetworkBehaviour.ArrayReader< T >	405
NetworkBehaviour.BehaviourReader< T >	406
NetworkBehaviour.ChangeDetector	407
NetworkBehaviour.ChangeDetector.Enumerable	410
NetworkBehaviour.ChangeDetector.Enumerator	411
NetworkBehaviour.DictionaryReader< K, V >	412
NetworkBehaviour.LinkListReader< T >	413
NetworkBehaviour.PropertyReader< T >	414
NetworkBehaviourBuffer	415
NetworkBehaviourBufferInterpolator	423
NetworkBehaviourUtils	438
NetworkBehaviourUtils.ArrayInitializer< T >	452
NetworkBehaviourUtils.DictionaryInitializer< K, V >	454
NetworkBehaviourUtils.MetaData	455
NetworkBehaviourWeavedAttribute	455
NetworkConfiguration	468
NetworkDeserializeMethodAttribute	473
NetworkDictionary< K, V >.Enumerator	480
NetworkDictionaryReadOnly< K, V >	481
NetworkedAttribute	484
NetworkedWeavedArrayAttribute	485
NetworkedWeavedAttribute	486
NetworkedWeavedDictionaryAttribute	487
NetworkedWeavedLinkedListAttribute	488
NetworkEvents.ConnectFailedEvent	491
NetworkEvents.ConnectRequestEvent	491
NetworkEvents.CustomAuthenticationResponse	491
NetworkEvents.DisconnectFromServerEvent	491
NetworkEvents.HostMigrationEvent	492
NetworkEvents.InputEvent	492
NetworkEvents.InputPlayerEvent	492
NetworkEvents.ObjectEvent	492
NetworkEvents.ObjectPlayerEvent	493
NetworkEvents.PlayerEvent	493
NetworkEvents.ReliableDataEvent	493
NetworkEvents.ReliableProgressEvent	493
NetworkEvents.RunnerEvent	494
NetworkEvents.SessionListUpdateEvent	494
NetworkEvents.ShutdownEvent	494
NetworkEvents.SimulationMessageEvent	494
NetworkId.EqualityComparer	501
NetworkInput	501
NetworkInputUtils	505
NetworkInputWeavedAttribute	506
NetworkLinkedList< T >.Enumerator	514
NetworkLinkedListReadOnly< T >	516
NetworkLoadSceneParameters	520
NetworkObjectFlagsExtensions	538
NetworkObjectGuid.EqualityComparer	549

NetworkObjectHeaderPtr	557
NetworkObjectMeta	559
NetworkObjectNestingKey.EqualityComparer	565
NetworkObjectReleaseContext	567
NetworkObjectSortKeyComparer	569
NetworkObjectSpawnException	570
NetworkObjectTypeId.EqualityComparer	581
NetworkPrefabAcquireContext	583
NetworkPrefabId.EqualityComparer	590
NetworkPrefabInfo	591
NetworkPrefabRef.EqualityComparer	600
NetworkPrefabTable	601
NetworkPrefabTableOptions	609
NetworkProjectConfig	610
NetworkProjectConfigAsset.SerializableSimulationBehaviourMeta	623
NetworkRpcStaticWeavedInvokerAttribute	629
NetworkRpcWeavedInvokerAttribute	630
NetworkRunnerCallbackArgs	698
NetworkRunnerCallbackArgs.ConnectRequest	698
NetworkRunnerUpdaterDefault.NetworkRunnerRender	701
NetworkRunnerUpdaterDefault.NetworkRunnerUpdate	702
NetworkRunnerUpdaterDefault.InvokeSettings	702
NetworkSceneAsyncOp	705
NetworkSceneAsyncOp.Awaiter	710
NetworkSceneLoadId	717
NetworkSceneObjectId	720
NetworkSerializeMethodAttribute	723
NetworkSimulationConfiguration	724
NetworkSpawnOp	728
NetworkSpawnOp.Awaiter	730
NetworkStructUtils	757
NetworkStructWeavedAttribute	758
NormalizedRectAttribute	768
OnChangedRenderAttribute	769
PreserveInPluginAttribute	779
Primes	780
PropertyAttribute	782
DecoratingPropertyAttribute	120
ArrayLengthAttribute	100
DisplayNameAttribute	125
DofAttributeBase	126
DrawIfAttribute	129
ErrorIfAttribute	156
WarnIfAttribute	1157
FieldEditorButtonAttribute	158
HideArrayElementLabelAttribute	189
InlineHelpAttribute	254
ReadOnlyAttribute	841
SerializeReferenceTypePickerAttribute	900
UnitAttribute	1124
DefaultForPropertyAttribute	122
DrawerPropertyAttribute	129
AssemblyNameAttribute	101
BinaryDataAttribute	112
BitSetAttribute	116
DisplayAsEnumAttribute	123
DrawInlineAttribute	131
ExpandableEnumAttribute	157

LayerAttribute	311
LayerMatrixAttribute	311
MaxStringByteCountAttribute	335
RangeExAttribute	839
ScenePathAttribute	879
ToggleLeftAttribute	1124
UnityAssetGuidAttribute	1126
UnityResourcePathAttribute	1133
FixedBufferPropertyAttribute	172
NetworkPrefabAttribute	585
ResolveNetworkPrefabSourceAttribute	860
ScriptHelpAttribute	887
SerializableTypeAttribute	899
UnityAddressablesRuntimeKeyAttribute	1125
BitStream	782
ICommunicator	824
IMessage	827
Ptr.EqualityComparer	833
ReadWriteUtils	842
ReadWriteUtilsForWeaver	846
ReflectionUtils	854
RenderAttribute	857
RenderTimeline	859
RenderWeavedAttribute	860
RpcAttribute	860
RpcHeader	863
RpcInfo	867
RpcInvokeData	870
RpcInvokeInfo	871
RpcSendResult	873
RpcTargetAttribute	874
RuntimeUnityFlagsSetup	875
SceneLoadDoneArgs	877
SerializableDictionary< TKey, TValue >	888
SerializableType< BaseType >	895
SessionInfo	901
Simulation.AreaOfInterest	922
SimulationBehaviourAttribute	927
SimulationBehaviourListScope	929
SimulationConfig	929
SimulationInput	934
SimulationInput.Buffer	936
SimulationInputHeader	940
SimulationMessage	941
SimulationMessagePtr	953
SimulationRuntimeConfig	953
INetBitWriteStream	955
NetBitBuffer	974
NetBitBufferNull	1002
INetPeerGroupCallbacks	958
Simulation	904
INetSocket	964
NetAddress	968
NetAddress.EqualityComparer	973
NetBitBuffer.Offset	999
NetBitBufferList	1000
NetBitBufferSerializer	1006
NetCommandAccepted	1011

NetCommandConnect	1011
NetCommandDisconnect	1012
NetCommandHeader	1013
NetCommandRefused	1014
NetConfig	1014
NetConfigNotify	1019
NetConfigSimulation	1021
NetConfigSimulationOscillator	1023
NetConnection	1026
NetConnectionId	1028
NetConnectionId.EqualityComparer	1030
NetConnectionMap	1031
NetConnectionMap.Iterator	1037
NetPeer	1038
NetPeerGroup	1046
NetSendEnvelope	1056
NetSocket	1057
ReliableHeader	1058
ReliableId	1059
ReliableKey	1062
ReliableList	1065
StunServers.StunServer	1068
StartGameArgs	1068
StartGameResult	1075
BehaviourStatisticsManager	1077
BehaviourStatisticsSnapshot	1078
FusionStatisticsManager	1079
FusionStatisticsSnapshot	1079
LagCompensationStatisticsSnapshot	1085
MemoryStatisticsSnapshot	1087
NetworkObjectStatisticsManager	1089
NetworkObjectStatisticsSnapshot	1090
Tick	1092
Tick.EqualityComparer	1099
Tick.RelationalComparer	1100
TickAccumulator	1101
TickRate	1105
TickRate.Resolved	1112
TickRate.Selection	1115
Timer	1120
TimeSyncConfiguration	1123
UnityContextMenuAttribute	1126
UnityDelayedAttribute	1127
UnityFormerlySerializedAsAttribute	1128
UnityHeaderAttribute	1128
UnityMinAttribute	1129
UnityMultilineAttribute	1130
UnityNonReorderableAttribute	1131
UnityNonSerializedAttribute	1131
UnityRangeAttribute	1132
UnitySerializeField	1134
UnitySerializeReference	1134
UnitySpaceAttribute	1134
UnityTooltipAttribute	1135
UTF32Tools	1136
UTF32Tools.CharEnumerator	1138
UTF32Tools.ConversionResult	1139
Versioning	1155

WeaverGeneratedAttribute	1159
IElementReaderWriter< bool >	228
IElementReaderWriter< K >	228
IElementReaderWriter< NetworkBehaviour >	228
NetworkBehaviour	383
IElementReaderWriter< NetworkObject >	228
NetworkBehaviour	383
IElementReaderWriter< V >	228
IExportedWordCount	1159
INetworkAssetSource< NetworkObject >	234
INetworkPrefabSource	241
IPublicFacingInterface	1160
IAfterAllTicks	214
NetworkMecanimAnimator	524
NetworkTransform	759
IAfterClientPredictionReset	216
IAfterHostMigration	216
IAfterRender	217
IAfterSpawned	218
IAfterTick	218
HitboxManager	193
IAfterUpdate	219
IAfterUpdateRemotePrefabs	219
IBeforeAllTicks	221
NetworkTransform	759
IBeforeClientPredictionReset	222
IBeforeCopyPreviousState	223
NetworkTransform	759
IBeforeHitboxRegistration	223
IBeforeSimulation	224
HitboxManager	193
IBeforeTick	225
IBeforeUpdate	225
IBeforeUpdateRemotePrefabs	226
IDespawned	227
NetworkBehaviour	383
IInputAuthorityGained	230
IInputAuthorityLost	231
IInterestEnter	231
IInterestExit	232
ILocalPrefabCreated	233
INetworkRunnerCallbacks	241
NetworkDelegates	472
NetworkEvents	489
IPlayerJoined	270
IPlayerLeft	271
IRemotePrefabCreated	271
ISceneLoadDone	272
ISceneLoadStart	273
ISimulationEnter	273
ISimulationExit	274
ISpawned	275
HitboxManager	193
NetworkBehaviour	383
IStateAuthorityChanged	275
NetworkedWeavedStringAttribute	1160
NetworkBehaviour.PropertyReader< Fusion.NetworkBehaviourId >	414
SerializableType< Fusion.SimulationBehaviour >	895

Chapter 4

Class Index

4.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

_128	A FixedStorage that can hold up to 128 words	73
_16	A FixedStorage that can hold up to 16 words	74
_2	A FixedStorage that can hold up to 2 words	75
_256	A FixedStorage that can hold up to 256 words	75
_32	A FixedStorage that can hold up to 32 words	76
_4	A FixedStorage that can hold up to 4 words	77
_512	A FixedStorage that can hold up to 512 words	78
_64	A FixedStorage that can hold up to 64 words	79
_8	A FixedStorage that can hold up to 8 words	80
Allocator		
	Memory Allocator	81
Allocator.Config		
	Memory Allocator Configuration	83
StaticConstructorAttribute		
	Attribute to mark a constructor as a static constructor	87
StaticFieldAttribute		
	Attribute to mark a static field with a reset mode	88
StaticFieldResetMethodAttribute		
	Attribute to mark a method as a static field reset method	89
Angle		
	A Networked fusion type for degrees. This can be used with the NetworkedAttribute , in RPCs, or in NetworkInput structs	90
ArrayLengthAttribute		
	Editor attribute for selecting the minimum and maximum length constraints for an array field	100
AssemblyNameAttribute		
	Specifies that the attributed field represents the name of an assembly	101

AssetObject	Base class for all Fusion assets	102
TaskManager	Task Factory is used to create new Tasks and Schedule long running Tasks	102
AtomicInt	Represents an atomic integer that provides thread-safe operations	106
AuthorityMasks	Provides constants and methods for managing authority masks	109
Behaviour	Alternative base class to Unity's MonoBehaviour. This allows for components that work both in Unity, as well as the Photon relays	110
BinaryDataAttribute	Specifies that the field represents binary data	112
BinUtils	Utility class for binary data	112
BitSetAttribute	Represents an attribute that specifies the number of bits in a bit set	116
CapacityAttribute	Capacity Attribute	117
CRC64	Provides methods to compute CRC64 checksums	118
DecoratingPropertyAttribute	A base class for property attributes that decorate other property attributes	120
DefaultForPropertyAttribute	Default For Property Attribute	122
DisplayAsEnumAttribute	Casts an enum or int value in the inspector to specific enum type for rendering of its popup list. Supplying a method name rather than a type allows a property with the type <code>Type</code> to be used to dynamically get the enum type	123
DisplayNameAttribute	Specifies the display name for a field	125
DofAttributeBase	Editor attribute for selective editor rendering. Condition member can be a property, field or method (with a return value)	126
DrawerPropertyAttribute	A base class for property attributes that are used to draw properties in the inspector	129
DrawIfAttribute	Editor attribute for selectively drawing/hiding fields. Condition member can be a property, field or method (with a return value)	129
DrawInlineAttribute	Specifies that a field should be drawn inline in the inspector	131
DynamicHeap	A dynamic heap for allocating and tracking unmanaged objects	131
DynamicHeap.Ignore	Ignore this field when scanning for pointers	135
DynamicHeap.Instance	Dynamic heap instance	135
EditorButtonAttribute	Specifies that a method should be displayed as a button in the Unity editor	139
DataEncryptor	Responsible for encrypting and decrypting data buffers	141
IDataEncryption	Interface for classes that manage the encryption/decryption of byte arrays	143
EncryptionConfig	Configuration for the Encryption Feature	146
EngineProfiler	Provides a set of methods to profile the engine	146

ErrorIfAttribute	Editor attribute for adding notices to fields if the condition member evaluates as <code>true</code> . Condition member can be a property, field or method (with a return value)	156
ExpandableEnumAttribute	Editor attribute that shows an enum as an expandable list of options in the inspector	157
FieldEditorButtonAttribute	Editor attribute to add a button that invokes a custom method in the inspector	158
FieldsMask< T >	Base class for FieldsMask<T>	160
FixedArray< T >	A fixed size array that can be used in structs	163
FixedArray< T >.Enumerator	Enumerator for the FixedArray struct	170
FixedBufferPropertyAttribute	Fixed Buffer Property Attribute	172
FixedStorage	Provides utility methods for fixed storage types	173
FloatCompressed	Represents a compressed float value for network transmission	174
FloatUtils	Provides utility methods for compressing and decompressing float values	177
FusionGlobalScriptableObject< T >	A base class for ScriptableObjects that are meant to be globally accessible, at edit-time and run-time. The way such objects are loaded is driven by usages of FusionGlobalScriptableObjectSourceAttribute attributes	179
FusionGlobalScriptableObjectAttribute	Provides additional information for a global scriptable object	182
FusionGlobalScriptableObjectLoadResult	The result of FusionGlobalScriptableObjectSourceAttribute.Load . Contains the loaded object and an optional unloader delegate	183
FusionGlobalScriptableObjectSourceAttribute	Base class for all attributes that can be used to load FusionGlobalScriptableObject . Attributes need to be registered at the assembly level. For instance, this snippet is used to register a default loader, that attempts to load from Resources based on FusionGlobalScriptableObjectAttribute.DefaultPath :	185
FusionMonoBehaviour	Base class for all Fusion MonoBehaviours	187
FusionScriptableObject	Base class for all Fusion scriptable objects	187
HeapConfiguration	Memory Heap Settings	187
HideArrayElementLabelAttribute	Attribute used to hide the label of an array element in the inspector	189
Hitbox	Represents a single lag-compensated collider. Multiple component instances can be added anywhere in the hierarchy of a NetworkObject which includes a HitboxRoot	189
HitboxManager	Entry point for lag compensated Hitbox queries, which maintains a history buffer, and provides lag compensated raycast and overlap methods. Singleton instance is accessible through the property <code>Runner.LagCompensation</code>	193
HitboxRoot	Root Hitbox group container. Manages registering/unregistering hitboxes with the group, and defines the broadphase geometry for the group	207
HostMigrationConfig	Project configuration settings specific to how the Host Migration behaves	213
HostMigrationToken	Transitory Holder with all necessary information to restart the Fusion Runner after the Host Migration has completed	214

IAfterAllTicks	Interface for AfterAllTicks callback. Called after the re-simulation loop (when applicable), and also after the forward simulation loop. Implement this interface on SimulationBehaviour and NetworkBehaviour classes	214
IAfterClientPredictionReset	Callback interface for AfterClientPredictionReset . Called at the very start of the resimulation loop (on clients with prediction enabled), immediately after state is set to the latest server snapshot. Implement this interface on SimulationBehaviour and NetworkBehaviour classes	216
IAfterHostMigration	Used to mark NetworkBehaviors that need to be react after a Host Migration process	216
IAfterRender	Interface for AfterRender callback. Called after the render loop	217
IAfterSpawned	Interface for AfterSpawned callback. Called after the object is spawned	218
IAfterTick	Interface for AfterTick callback. Called after each tick simulation completes. Implement this interface on SimulationBehaviour and NetworkBehaviour classes	218
IAfterUpdate	Interface for the AfterUpdate callback, which is called at the end of each Fusion Update segment. Implement this interface on SimulationBehaviour and NetworkBehaviour classes	219
IAfterUpdateRemotePrefabs	Invoked after updating remote prefabs	219
IAsyncOperation	Defines an asynchronous operation	220
IBeforeAllTicks	Interface for BeforeAllTicks callback. Called before the re-simulation loop (when applicable), and also before the forward simulation loop. Implement this interface on SimulationBehaviour and NetworkBehaviour classes	221
IBeforeClientPredictionReset	Callback interface for BeforeClientPredictionReset . Called at the very start of the re-simulation loop (on clients with prediction enabled), before state is set to the latest server snapshot. Implement this interface on SimulationBehaviour and NetworkBehaviour classes	222
IBeforeCopyPreviousState	Interface for BeforeCopyPreviousState callback. Called before the copy of the previous state	223
IBeforeHitboxRegistration	Interface for BeforeHitboxRegistration callback. Implement this interface on SimulationBehaviour and NetworkBehaviour classes	223
IBeforeSimulation	Interface for BeforeSimulation callback. Called before both the re-simulation (when applicable) and forward simulation loops. Implement this interface on SimulationBehaviour and NetworkBehaviour classes	224
IBeforeTick	Interface for BeforeTick callback. Called before each tick is simulated. Implement this interface on SimulationBehaviour and NetworkBehaviour classes	225
IBeforeUpdate	Interface for the BeforeUpdate callback, which is called at the beginning of each Fusion Update segment. Implement this interface on SimulationBehaviour and NetworkBehaviour classes	225
IBeforeUpdateRemotePrefabs	Invoked before updating remote prefabs	226
ICoroutine	Defines a coroutine	227
IDespawned	Interface for Despawned callback. Called when a NetworkBehaviour is despawned	227
IElementReaderWriter< T >	Defines the interface for reading and writing elements in a byte array	228
IFixedStorage	Interface for fixed storage types	230

IInputAuthorityGained	Interface for handling the event when the input authority is gained	230
IInputAuthorityLost	Interface for handling the event when the input authority is lost	231
IInterestEnter	Interface for handling the event when a player enters the area of interest	231
IInterestExit	Interface for handling the event when a player exits the area of interest	232
ILocalPrefabCreated	Interface for handling the event when a local prefab is created	233
INetworkArray	Defines the interface for a networked array	233
INetworkAssetSource< T >	Interface for a network asset source	234
INetworkDictionary	Defines the interface for a networked dictionary	236
INetworkInput	Flag interface for custom NetworkInput structs	237
INetworkLinkedList	Defines the interface for a networked linked list	237
INetworkObjectInitializer	Interface for initializing network objects	238
INetworkObjectProvider	Interface which defines the handlers for NetworkRunner Spawn() and Despawn() actions. Passing an instance of this interface to NetworkRunner.StartGame(StartGameArgs) as the StartGameArgs.ObjectProvider argument value will assign that instance as the handler for runner Spawn() and Despawn() actions. By default (if StartGameArgs.ObjectProvider == null) actions will use Instantiate() , and Despawn() actions will use Destroy()	238
INetworkPrefabSource	Interface for a network prefab source	241
INetworkRunnerCallbacks	Interface for NetworkRunner callbacks. Register a class/struct instance which implements this interface with NetworkRunner.AddCallbacks(INetworkRunnerCallbacks[])	241
INetworkRunnerUpdater	Interface which defines the handlers for NetworkRunner Updates. An implementation is responsible for calling NetworkRunner.UpdateInternal(double) and NetworkRunner.RenderInternal periodically	248
INetworkSceneManager	Interface for a NetworkRunner scene manager. A scene manager is responsible for loading and unloading scenes	249
INetworkStruct	Base interface for all Fusion Network Structs	253
INetworkTRSPTeleport	Implement this interface on a NetworkTRSP implementation to indicate it can be teleported	254
InlineHelpAttribute	If applied to a field, checks if there is a help text for the field or the field's type and shows it in the inspector	254
IUnitySurrogate	Represents an interface for Unity surrogates. This interface provides methods for reading and writing data	255
IUnityValueSurrogate< T >	Represents an interface for Unity value surrogates. This interface provides a property for accessing and modifying the data	256
UnityArraySurrogate< T, ReaderWriter >	A base class for Unity array surrogates	257
UnityDictionarySurrogate< TKeyType, TKeyReaderWriter, TValueType, TValueReaderWriter >	A surrogate for serializing a dictionary	259

UnityLinkedListSurrogate< T, ReaderWriter >	
A surrogate for serializing a linked list	261
UnitySurrogateBase	
Represents a base class for Unity surrogates. This class is serializable and provides abstract methods for reading, writing, and initializing data	263
UnityValueSurrogate< T, TReaderWriter >	
Represents a base class for Unity value surrogates. This class is serializable and provides methods for reading, writing, and initializing data	265
InterpolatedErrorCorrectionSettings	
A set of parameters that tune the interpolated correction of prediction error on transform data	267
IPlayerJoined	
Interface for handling the event when a player joins the game	270
IPlayerLeft	
Interface for handling the event when a player leaves the game	271
IRemotePrefabCreated	
Interface for handling the event when a remote prefab is created	271
ISceneLoadDone	
Interface for handling the event when a scene load operation is completed	272
ISceneLoadStart	
Interface for handling the event when a scene load operation is started	273
ISimulationEnter	
Interface for SimulationEnter callback. Called when the NetworkObject joins AreaOfInterest. Implement this interface on SimulationBehaviour and NetworkBehaviour classes	273
ISimulationExit	
Interface for the SimulationExit callback. Called when the NetworkObject leaves AreaOfInterest. Implement this interface on SimulationBehaviour and NetworkBehaviour classes	274
ISpawned	
Interface for handling the event when an object is spawned	275
IStateAuthorityChanged	
Interface for handling the event when the state authority changes	275
LagCompensatedHit	
Defines a lag compensated query hit result	276
AABB	
Represents an Axis-Aligned Bounding Box (AABB)	279
BoxOverlapQuery	
Class that represents a box overlap query. Used to query against the NetworkRunner.LagCompensation API	281
BoxOverlapQueryParams	
Base parameters needed to execute a box overlap query	284
BVHDraw	
Provide a way to iterate over BVH and return a BVHNodeDrawInfo for each node	286
BVHNodeDrawInfo	
Container class to provide the necessary info to draw nodes from the BVH	286
ColliderDrawInfo	
Container class to provide the necessary information to draw a hitbox collider	287
HitboxColliderContainerDraw	
Provide a way to iterate over the HitboxBuffer.HitboxSnapshot and return the ColliderDrawInfo for each collider on the snapshot	289
LagCompensatedExt	
LagCompensated Extension methods	290
LagCompensationDraw	
Provide access to iterate over the lag compensation system components and give the necessary information to draw them	291
LagCompensationUtils.ContactData	
Details regarding a shape intersection. It does not carry information about the intersection happening or not	292
PositionRotationQueryParams	
Query parameters for position rotation query	293

Query	Base class for all Lag Compensation queries	294
QueryParams	Base parameters needed to execute a query	298
RaycastAllQuery	Class that represents a raycast all query. Used to query against the NetworkRunner.LagCompensation API	300
RaycastQuery	Class that represents a raycast query. Used to query against the NetworkRunner.LagCompensation API	301
RaycastQueryParams	Base parameters needed to execute a raycast query	303
SnapshotHistoryDraw	Provide a way to iterate over the HitboxBuffer and return the HitboxColliderContainerDraw container for each snapshot on the buffer	305
SphereOverlapQuery	Class that represents a sphere overlap query. Used to query against the NetworkRunner.LagCompensation API	305
SphereOverlapQueryParams	Base parameters needed to execute a sphere overlap query	307
LagCompensationSettings	Settings for lag compensation history	309
LayerAttribute	Specifies that an int field should be drawn as a layer field in the inspector	311
LayerMatrixAttribute	Specifies that the integer array field should be drawn as a layer matrix in the inspector	311
LobbyInfo	Holds information about a Lobby	311
Mask256	Mask256 is a 256-bit mask that can be used to store 256 boolean values	312
Maths	Math utility methods	317
Maths.FastAbs	Represents a structure that allows quick setting of the sign bit of floats using field offsets	334
MaxStringByteCountAttribute	Specifies that the string field should be drawn as a text field with a maximum byte count for given encoding	335
Native	Native Memory Allocator	336
NestedComponentUtilities	Tools to replace GetComponent variants that respects nested objects. These are used to find components of a NetworkedObjects without also finding components that belong to parent or child NetworkedObjects	364
NetworkArray< T >	Fusion type for networking arrays. Maximum capacity is fixed, and is set with the CapacityAttribute .	372
NetworkArray< T >.Enumerator	Enumerator for NetworkArray	379
NetworkArrayExtensions	Provides extension methods for the NetworkArray class	381
NetworkArrayReadOnly< T >	Provides a read-only view of a network array	382
NetworkAssemblyIgnoreAttribute	Network Assembly Ignore Attribute	383
NetworkAssemblyWeavedAttribute	Network Assembly Weaved Attribute	383

NetworkBehaviour	Base class for Fusion network components, which are associated with a NetworkObject	383
NetworkBehaviour.ArrayReader< T >	Provides a reader for network arrays of type T	405
NetworkBehaviour.BehaviourReader< T >	Provides a reader for network behaviours of type T	406
NetworkBehaviour.ChangeDetector	Change detector for a NetworkBehaviour	407
NetworkBehaviour.ChangeDetector.Enumerable	Struct representing a collection of changes detected in a NetworkBehaviour	410
NetworkBehaviour.ChangeDetector.Enumerator	Enumerator for the collection of changes detected in a NetworkBehaviour	411
NetworkBehaviour.DictionaryReader< K, V >	Provides a reader for network dictionaries with keys of type K and values of type V	412
NetworkBehaviour.LinkListReader< T >	Provides a reader for network linked lists of type T	413
NetworkBehaviour.PropertyReader< T >	Provides a reader for properties of type T in a network behaviour	414
NetworkBehaviourBuffer	Provides low level accesss to data buffers that can be read using a NetworkBehaviour.Reader	415
NetworkBehaviourBufferInterpolator	The NetworkBehaviourBufferInterpolator struct is used to interpolate between two NetworkBehaviourBuffer instances. This is a read-only, ref struct, meaning it cannot be boxed and it can only be used on the stack	423
NetworkBehaviourId	Represents the unique identifier for a NetworkBehaviour instance	434
NetworkBehaviourUtils	This static class provides utility methods for working with NetworkBehaviour objects	438
NetworkBehaviourUtils.ArrayInitializer< T >	A utility structure for initializing NetworkArray and NetworkLinkedList with inline initialization	452
NetworkBehaviourUtils.DictionaryInitializer< K, V >	A utility structure for initializing NetworkDictionary with inline initialization	454
NetworkBehaviourUtils.MetaData	This structure holds metadata for a NetworkBehaviour object	455
NetworkBehaviourWeavedAttribute	Network Behaviour Weaved Attribute	455
NetworkBool	Represents a boolean value that can be networked	456
NetworkButtons	Represents a set of buttons that can be networked	459
NetworkConfiguration	Main network configuration class	468
NetworkDelegates	Network Runner Callbacks Delegates	472
NetworkDeserializeMethodAttribute	Network Deserialize Method Attribute	473
NetworkDictionary< K, V >	Fusion type for networking Dictionaries. Maximum capacity is fixed, and is set with the CapacityAttribute . 473	
NetworkDictionary< K, V >.Enumerator	Enumerator for NetworkDictionary	480
NetworkDictionaryReadOnly< K, V >	A read-only version of NetworkDictionary<TKey,TValue>	481
NetworkedAttribute	484	

NetworkedWeavedArrayAttribute	Attribute applied to an array property by the weaver	485
NetworkedWeavedAttribute	Networked Weaved Attribute	486
NetworkedWeavedDictionaryAttribute	Attribute applied to a dictionary property by the weaver	487
NetworkedWeavedLinkedListAttribute	Attribute applied to a list property by the weaver	488
NetworkEvents	Companion component for NetworkRunner . Exposes INetworkRunnerCallbacks as UnityEvents, which can be wired up to other components in the inspector	489
NetworkEvents.ConnectFailedEvent	UnityEvent for ConnectFailed	491
NetworkEvents.ConnectRequestEvent	UnityEvent for ConnectRequest	491
NetworkEvents.CustomAuthenticationResponse	UnityEvent for Custom Authentication	491
NetworkEvents.DisconnectFromServerEvent	UnityEvent for DisconnectFromServer	491
NetworkEvents.HostMigrationEvent	UnityEvent for HostMigration	492
NetworkEvents.InputEvent	UnityEvent for NetworkInput	492
NetworkEvents.InputPlayerEvent	UnityEvent for NetworkInput with PlayerRef	492
NetworkEvents.ObjectEvent	UnityEvent for NetworkObject	492
NetworkEvents.ObjectPlayerEvent	UnityEvent for NetworkObject with PlayerRef	493
NetworkEvents.PlayerEvent	UnityEvent for PlayerRef	493
NetworkEvents.ReliableDataEvent	UnityEvent for Reliable Data	493
NetworkEvents.ReliableProgressEvent	UnityEvent for Reliable Data Progress	493
NetworkEvents.RunnerEvent	UnityEvent for NetworkRunner	494
NetworkEvents.SessionListUpdateEvent	UnityEvent for SessionInfo List	494
NetworkEvents.ShutdownEvent	UnityEvent for Shutdown	494
NetworkEvents.SimulationMessageEvent	UnityEvent for SimulationMessage	494
NetworkId	The unique identifier for a network entity	495
NetworkId.EqualityComparer	IEqualityComparer interface for NetworkId objects	501
NetworkInput	NetworkInput Struct	501
NetworkInputUtils	Utility methods for NetworkInput	505
NetworkInputWeavedAttribute	Network Input Weaved Attribute	506
NetworkLinkedList< T >	Fusion type for networking LinkedLists. Maximum capacity is fixed, and is set with the CapacityAttribute . Typical Usage:	507

NetworkLinkedList< T >.Enumerator	
Enumerator for NetworkLinkedList<T>	514
NetworkLinkedListReadOnly< T >	
Read-only version of NetworkLinkedList<T>	516
NetworkLoadSceneParameters	
Parameters for loading a scene	520
NetworkMecanimAnimator	
A component for synchronizing the Animator controller state from the State Authority to network proxies. Requires a Unity Animator component, and a NetworkObject component. NOTE↔ : Animator Root Motion is not compatible with re-simulation and prediction	524
NetworkObject	
The primary Fusion component for networked GameObject entities. This stores the object's network identity and manages the object's state and input authority	527
NetworkObjectFlagsExtensions	
Extension methods for the NetworkObjectFlags enum	538
NetworkObjectGuid	
NetworkObjectGuid	540
NetworkObjectGuid.EqualityComparer	
EqualityComparer for NetworkObjectGuid	549
NetworkObjectHeader	
Network object header information for a NetworkObject	550
NetworkObjectHeaderPtr	
Represents a pointer to a NetworkObjectHeader . This struct is unsafe because it uses pointers	557
NetworkObjectInitializerUnity	
Initializes network objects for Unity	559
NetworkObjectMeta	
Meta information about a network object	559
NetworkObjectNestingKey	
A key used to identify a network object nesting	561
NetworkObjectNestingKey.EqualityComparer	
Implements the IEqualityComparer interface	565
NetworkObjectPrefabData	
This class represents the data for a network object prefab	566
NetworkObjectProviderDummy	
A dummy implementation of the INetworkObjectProvider interface. This class is used for testing purposes and throws a NotImplementedException for all its methods	567
NetworkObjectReleaseContext	
Represents the context for releasing a network object. This struct is unsafe because it uses pointers	567
NetworkObjectSortKeyComparer	
This class is used to compare two NetworkObject instances based on their SortKey . It implements the IComparer interface	569
NetworkObjectSpawnException	
Network Object Spawn Exception	570
NetworkObjectTypeId	
ID for a NetworkObject Prefab which has been cataloged in a NetworkProjectConfig.PrefabTable	572
NetworkObjectTypeId.EqualityComparer	
NetworkObjectTypeId Comparer	581
NetworkPhysicsInfo	
Network Physics INetworkStruct	582
NetworkPrefabAcquireContext	
Represents the context for acquiring a prefab instance for a network object. This struct is unsafe because it uses pointers	583
NetworkPrefabAttribute	
Network Prefab Attribute	585
NetworkPrefabId	
ID for a NetworkObject Prefab which has been cataloged in a NetworkProjectConfig.PrefabTable	585

NetworkPrefabId.EqualityComparer	
Equality comparer for NetworkPrefabId	590
NetworkPrefabInfo	
Meta data for a NetworkObject prefab which has been cataloged in a NetworkProjectConfig.PrefabTable	591
NetworkPrefabRef	
NetworkPrefabRef	593
NetworkPrefabRef.EqualityComparer	
EqualityComparer for NetworkPrefabRef	600
NetworkPrefabTable	
Class representing a table of network prefabs	601
NetworkPrefabTableOptions	
Options for the NetworkPrefabTable	609
NetworkProjectConfig	
The core Fusion config file that is shared with all peers at startup	610
NetworkProjectConfigAsset	
Manages and references the current instance of NetworkProjectConfig	620
NetworkProjectConfigAsset.SerializableSimulationBehaviourMeta	
An auto-generated list containing meta information about all the SimulationBehaviours in the project, e.g. execution order	623
NetworkRNG	
PCG32 random generator, 16 bytes in size. http://www.pcg-random.org	624
NetworkRpcStaticWeavedInvokerAttribute	
Network Rpc Static Weaved Invoker Attribute Contains info about a static weaved RPC Method	629
NetworkRpcWeavedInvokerAttribute	
Network Rpc Weaved Invoker Attribute Contains info about a weaved RPC Method	630
NetworkRunner	
Host Migration related code in order to get a copy of the Simulation State	631
NetworkRunnerCallbackArgs	
Stores data types used on the INetworkRunnerCallbacks interface	698
NetworkRunnerCallbackArgs.ConnectRequest	
Data holder of a Connection Request from a remote client	698
NetworkRunnerUpdaterDefault	
Default implementation of INetworkRunnerUpdater that uses the Unity PlayerLoop	699
NetworkRunnerUpdaterDefault.NetworkRunnerRender	
Used to invoke NetworkRunner.RenderInternal in the PlayerLoop	701
NetworkRunnerUpdaterDefault.NetworkRunnerUpdate	
Used to invoke NetworkRunner.UpdateInternal(double) in the PlayerLoop	702
NetworkRunnerUpdaterDefault.InvokeSettings	
Settings for the NetworkRunnerUpdaterDefault	702
NetworkSceneAsyncOp	
A wrapper for async scene operations	705
NetworkSceneAsyncOp.Awaiter	
Awaiter for NetworkSceneAsyncOp	710
NetworkSceneInfo	
Can store up to 8 active scenes and allows for duplicates. Each write increases Version which can be used to generate unique scene objects ids for when a scene is supposed to be reloaded	711
NetworkSceneLoadId	
A unique identifier for a scene load operation	717
NetworkSceneObjectId	
A unique identifier for a scene object	720
NetworkSerializeMethodAttribute	
Network Serialize Method Attribute	723
NetworkSimulationConfiguration	
Configuration for network conditions simulation (induced latency and loss)	724
NetworkSpawnOp	
Spawn Operation	728

NetworkSpawnOp.Awaiter	
Awaiter for NetworkSpawnOp	730
NetworkString< TSize >	
Fixed-size UTF32 string. All operations are alloc-free, except for converting to System.String	732
NetworkStructUtils	
Utility methods for INetworkStruct	757
NetworkStructWeavedAttribute	
Describes the total number of WORDs a INetworkStruct uses	758
NetworkTransform	
Add to any NetworkObject Transform, or its associated child Transforms to automatically synchronize TRSP (Position/Rotation/Scale/Parent)	759
NetworkTRSP	
Base class for spatial (Position/Rotation/Scale/Parent) synchronization component, such as NetworkTransform . Provides the base logic for render interpolation, parenting synchronization, and teleport, that can be used in components derived from this class	762
NetworkTRSPData	
Data structure storing spatial (Position/Rotation/Scale/Parent) synchronization data for spatial synchronization components, NetworkTRSP and its subclass NetworkTransform	765
NormalizedRectAttribute	
Enables a special inspector drawer for Unity Rect type, specially designed for editing Rect↔ Transforms using normalized values	768
OnChangedRenderAttribute	
OnChangedRender Attribute	769
PlayerRef	
Represents a Fusion player	771
PreserveInPluginAttribute	
Preserve In Plugin Attribute	779
Primes	
Provides a set of methods to work with prime numbers	780
PropertyAttribute	
Specifies that the attribute can be applied to fields only	782
BitStream	
BitStream serialization methods	782
ICommunicator	
Interface for a Communicator	824
IMessage	
Represents a Protocol Message	827
Ptr	
Ptr	828
Ptr.EqualityComparer	
Ptr Equality Comparer	833
QuaternionCompressed	
Represents a compressed Quaternion value for network transmission	834
RangeExAttribute	
Represents an attribute that specifies a range of values for a field or property	839
ReadOnlyAttribute	
Attribute used to mark a field as read-only	841
ReadWriteUtils	
Provides utility methods for reading and writing data	842
ReadWriteUtilsForWeaver	
Provides utility methods for reading and writing data	846
ReflectionUtils	
Provides utility methods for reflection	854
RenderAttribute	
Override default render settings for [Networked] properties	857
RenderTimeline	
Can be used to acquire interpolated data for different points in time	859

RenderWeavedAttribute	
Render Weaved Attribute	860
ResolveNetworkPrefabSourceAttribute	
Resolve Network Prefab Source Attribute	860
RpcAttribute	
Flags a method as being a networked Remote Procedure Call. Only usable in a NetworkBehaviour . Calls to this method (from the indicated allowed RpcSources) will generate a network message, which will execute the method remotely on the indicated RpcTargets . The RPC method can include an empty RpcInfo argument, that will include meta information about the RPC on the receiving peer	860
RpcHeader	
Header for RPC messages	863
RpcInfo	
RpcInfo is a struct that contains information about the RPC message	867
RpcInvokeData	
Represents the data required to invoke an RPC message	870
RpcInvokeInfo	
May be used as an optional RpcAttribute return value. Contains meta data about the RPC send, such as failure to send reasons, culling, message size, etc	871
RpcSendResult	
RPC send operation result information	873
RpcTargetAttribute	
RPC attribute used to indicate a specific target player for an RPC when sending from one player to another. RPC is sent to the server, and then is forwarded to the specified player. Usage: . . .	874
RuntimeUnityFlagsSetup	
Contains methods to setup runtime flags for Unity	875
SceneLoadDoneArgs	
Struct that contains information about a scene after it has been loaded	877
ScenePathAttribute	
Specifies that a string field represents a scene path	879
SceneRef	
Scene reference struct. Can be used to reference a scene by index or by path	879
ScriptHelpAttribute	
Defines the appearance of the script header in the Unity inspector	887
SerializableDictionary< TKey, TValue >	
A serializable dictionary	888
SerializableType< BaseType >	
A System.Type wrapper that can be serialized	895
SerializableTypeAttribute	
Specifies that either a string field represents a type name or sets additional options for SerializableType field	899
SerializeReferenceTypePickerAttribute	
Attribute used to show a type picker for a field with [SerializeReference]	900
SessionInfo	
Holds information about the Game Session	901
Simulation	
Main simulation class	904
Simulation.AreaOfInterest	
Area of Interest Definition	922
SimulationBehaviour	
Base class for a Fusion aware Behaviour (derived from UnityEngine.MonoBehaviour). If a SimulationBehaviour is found on a NetworkRunner game object during the runner initialisation, the SimulationBehaviour is automatically registered. Objects derived from this object can be associated with a NetworkRunner and Simulation using NetworkRunner.AddGlobal()	925
SimulationBehaviourAttribute	
Attribute for specifying which SimulationStages and SimulationModes this SimulationBehaviour will execute in. Can be used to limit execution to only Host, Server or Client peers, or to only execute on Resimulation or Forward ticks. Usage:	927

SimulationBehaviourListScope	Provides a scope for a SimulationBehaviourUpdater.BehaviourList , incrementing its lock count on creation and decrementing it on disposal. If the lock count reaches zero on disposal, all pending removals in the list are performed	929
SimulationConfig	Project configuration settings specific to how the Simulation class behaves	929
SimulationInput	Simulation Input	934
SimulationInput.Buffer	Buffer for SimulationInputs	936
SimulationInputHeader	Simulation Input Header	940
SimulationMessage	Simulation Message	941
SimulationMessagePtr	Simulation Message Pointer	953
SimulationRuntimeConfig	Stores the runtime configuration of the simulation	953
INetBitWriteStream	Interface for writing bits to a stream	955
INetPeerGroupCallbacks	Defines the callbacks for network peer group events	958
INetSocket	Defines the interface for network socket operations	964
NetAddress	Represents a Network Address, which includes a IP and Port This can contains either a IPv4 or a IPv6 address	968
NetAddress.EqualityComparer	Provides methods to compare two NetAddress instances for equality	973
NetBitBuffer	Represents a buffer for reading and writing bits	974
NetBitBuffer.Offset	Represents an offset within a NetBitBuffer	999
NetBitBufferList	Represents a linked list of Fusion.Sockets.NetBitBuffer	1000
NetBitBufferNull	Represents a null bit buffer for writing data	1002
NetBitBufferSerializer	Represents a serializer for reading and writing data to a NetBitBuffer	1006
NetCommandAccepted	Accepted Command, sent by the server when a remote client connection is accepted	1011
NetCommandConnect	Connect Command used to signal a remote server that a client is trying to connect to it	1011
NetCommandDisconnect	Disconnect Command, it can be used by either side of the connection	1012
NetCommandHeader	Network Command Header Describe its type and usual settings for all commands	1013
NetCommandRefused	Refuse Command, sent by the server when the connection was refused. This happens when the server has reached its max connection capacity	1014
NetConfig	General configuration used to drive the behavior of the Socket library	1014
NetConfigNotify	Represents the configuration for network notifications	1019
NetConfigSimulation	Represents the configuration for network simulation	1021
NetConfigSimulationOscillator	Represents an oscillator configuration for network simulation	1023

NetConnection	
Network connection	1026
NetConnectionId	
Represents a network connection ID	1028
NetConnectionId.EqualityComparer	
An equality comparer for NetConnectionId instances	1030
NetConnectionMap	
Represents a network connection map	1031
NetConnectionMap.Iterator	
Iterator for traversing the connections in a NetConnectionMap	1037
NetPeer	
Network Peer	1038
NetPeerGroup	
Network Peer Group	1046
NetSendEnvelope	
Represents an envelope for sending network packets in the Fusion.Sockets namespace	1056
NetSocket	
Represents a network socket with a handle and a native socket	1057
ReliableHeader	
Represents a reliable header structure used in the Fusion.Sockets namespace	1058
ReliableId	
Represents a reliable identifier used in the Fusion.Sockets namespace	1059
ReliableKey	
Represents a reliable key structure used in the Fusion.Sockets namespace	1062
ReliableList	
Represents a list of reliable headers	1065
StunServers.StunServer	
Stores Addresses of a STUN Server	1068
StartGameArgs	
Fusion Start Arguments, used to configure the simulation mode and other settings	1068
StartGameResult	
Represents the result of starting the Fusion Simulation	1075
BehaviourStatisticsManager	
Behaviour statistics manager will provide access to the behaviour statistics snapshots	1077
BehaviourStatisticsSnapshot	
Represents a snapshot of statistics related to Fusion Behaviour type execution	1078
FusionStatisticsManager	
Represents a fusion statistics manager	1079
FusionStatisticsSnapshot	
Represents a snapshot of Fusion statistics	1079
LagCompensationStatisticsSnapshot	
Represents a snapshot of lag compensation statistics	1085
MemoryStatisticsSnapshot	
Represents a snapshot of specific memory statistics. For basic total allocated memory and total free memory check the Fusion Statistics	1087
NetworkObjectStatisticsManager	
Manages network object statistics for monitored network objects	1089
NetworkObjectStatisticsSnapshot	
Represents a snapshot of network object statistics	1090
Tick	
A tick is a 32-bit integer that represents a frame number	1092
Tick.EqualityComparer	
Provides a mechanism for comparing two Tick objects for equality	1099
Tick.RelationalComparer	
Provides a mechanism for comparing two Tick objects	1100
TickAccumulator	
A tick accumulator	1101

TickRate	A tick rate is a collection of tick rates	1105
TickRate.Resolved	Represents a resolved tick rate	1112
TickRate.Selection	Represents a selection of tick rates for client and server	1115
TickTimer	A timer that is based on ticks instead of seconds	1116
Timer	Represents a high-resolution timer	1120
TimeSyncConfiguration	Time Synchronization Configuration	1123
ToggleLeftAttribute	Specifies that the bool field should be drawn as the toggle on the left side of the label	1124
UnitAttribute	Unit Attribute class. Used to mark a field with the respective Units	1124
UnityAddressablesRuntimeKeyAttribute	Specifies that the string field represents a key for Unity Addressables	1125
UnityAssetGuidAttribute	Specifies that the string field represents a GUID of an asset	1126
UnityContextMenuAttribute	Unity ContextMenuItemAttribute	1126
UnityDelayedAttribute	Unity DelayedAttribute	1127
UnityFormerlySerializedAsAttribute	Unity FormerlySerializedAsAttribute	1128
UnityHeaderAttribute	Unity HeaderAttribute	1128
UnityMinAttribute	Unity MinAttribute	1129
UnityMultilineAttribute	Unity MultilineAttribute	1130
UnityNonReorderableAttribute	Unity NonReorderableAttribute	1131
UnityNonSerializedAttribute	Unity NonSerializedAttribute	1131
UnityRangeAttribute	Unity RangeAttribute	1132
UnityResourcePathAttribute	Specifies that the string field represents a path to a Unity resource	1133
UnitySerializeField	Unity SerializeField	1134
UnitySerializeReference	Unity SerializeReference	1134
UnitySpaceAttribute	Unity SpaceAttribute	1134
UnityTooltipAttribute	Unity TooltipAttribute	1135
UTF32Tools	UTF32Tools provides a set of methods to work with UTF32 encoded strings	1136
UTF32Tools.CharEnumerator	Enumerates the characters in a UTF-32 encoded string	1138
UTF32Tools.ConversionResult	Represents the result of a conversion operation, containing the number of characters and code points processed	1139
Vector2Compressed	Represents a compressed Vector2 value for network transmission	1140

Vector3Compressed	
Represents a compressed Vector3 value for network transmission	1145
Vector4Compressed	
Represents a compressed Vector4 value for network transmission	1150
Versioning	
The Versioning class provides methods and properties related to versioning	1155
WarnIfAttribute	
Editor attribute for adding notices to fields if the condition member evaluates as <code>true</code> . Condition member can be a property, field or method (with a return value)	1157
WeaverGeneratedAttribute	
Weaver Generated Attribute	1159
IExportedWordCount	
Used in plugin. Indicates that <code>NetworkBehaviour.DynamicWordCount</code> is exported and can be assigned from serialized data	1159
IPublicFacingInterface	
Tag Interface for all public facing Fusion interfaces	1160
NetworkedWeavedStringAttribute	
An attribute emitted by the weaver to mark a string field as networked	1160

Chapter 5

Namespace Documentation

5.1 Fusion Namespace Reference

Classes

- struct [_128](#)
A [FixedStorage](#) that can hold up to 128 words.
- struct [_16](#)
A [FixedStorage](#) that can hold up to 16 words.
- struct [_2](#)
A [FixedStorage](#) that can hold up to 2 words.
- struct [_256](#)
A [FixedStorage](#) that can hold up to 256 words.
- struct [_32](#)
A [FixedStorage](#) that can hold up to 32 words.
- struct [_4](#)
A [FixedStorage](#) that can hold up to 4 words.
- struct [_512](#)
A [FixedStorage](#) that can hold up to 512 words.
- struct [_64](#)
A [FixedStorage](#) that can hold up to 64 words.
- struct [_8](#)
A [FixedStorage](#) that can hold up to 8 words.
- struct [Allocator](#)
Memory [Allocator](#)
- struct [Angle](#)
A [Networked](#) fusion type for degrees. This can be used with the [NetworkedAttribute](#), in [RPCs](#), or in [NetworkInput](#) structs.
- class [ArrayLengthAttribute](#)
Editor attribute for selecting the minimum and maximum length constraints for an array field.
- class [AssemblyNameAttribute](#)
Specifies that the attributed field represents the name of an assembly.
- class [AssetObject](#)
Base class for all [Fusion](#) assets.
- struct [AtomicInt](#)
Represents an atomic integer that provides thread-safe operations.

- class [AuthorityMasks](#)
Provides constants and methods for managing authority masks.
- class [Behaviour](#)
Alternative base class to Unity's `MonoBehaviour`. This allows for components that work both in Unity, as well as the Photon relays.
- class [BinaryDataAttribute](#)
Specifies that the field represents binary data.
- class [BinUtils](#)
Utility class for binary data.
- class [BitSetAttribute](#)
Represents an attribute that specifies the number of bits in a bit set.
- class [CapacityAttribute](#)
Capacity Attribute
- class [CRC64](#)
Provides methods to compute `CRC64` checksums.
- class [DecoratingPropertyAttribute](#)
A base class for property attributes that decorate other property attributes.
- class [DefaultForPropertyAttribute](#)
Default For Property Attribute
- class [DisplayAsEnumAttribute](#)
Casts an enum or int value in the inspector to specific enum type for rendering of its popup list. Supplying a method name rather than a type allows a property with the type `Type` to be used to dynamically get the enum type.
- class [DisplayNameAttribute](#)
Specifies the display name for a field.
- class [DolfAttributeBase](#)
Editor attribute for selective editor rendering. Condition member can be a property, field or method (with a return value).
- class [DrawerPropertyAttribute](#)
A base class for property attributes that are used to draw properties in the inspector.
- class [DrawIfAttribute](#)
Editor attribute for selectively drawing/hiding fields. Condition member can be a property, field or method (with a return value).
- class [DrawInlineAttribute](#)
Specifies that a field should be drawn inline in the inspector.
- struct [DynamicHeap](#)
A dynamic heap for allocating and tracking unmanaged objects.
- class [DynamicHeapInstance](#)
Dynamic heap instance.
- class [EditorButtonAttribute](#)
Specifies that a method should be displayed as a button in the Unity editor.
- class [EncryptionConfig](#)
Configuration for the `Encryption` Feature
- class [EngineProfiler](#)
Provides a set of methods to profile the engine.
- class [ErrorIfAttribute](#)
Editor attribute for adding notices to fields if the condition member evaluates as `true`. Condition member can be a property, field or method (with a return value).
- class [ExpandableEnumAttribute](#)
Editor attribute that shows an enum as an expandable list of options in the inspector.
- class [FieldEditorButtonAttribute](#)
Editor attribute to add a button that invokes a custom method in the inspector.
- class [FieldsMask](#)

- Base class for [FieldsMask<T>](#).*
- class [FixedArray](#)

A fixed size array that can be used in structs.
 - class [FixedBufferPropertyAttribute](#)

Fixed Buffer Property Attribute
 - class [FixedStorage](#)

Provides utility methods for fixed storage types.
 - struct [FloatCompressed](#)

Represents a compressed float value for network transmission.
 - class [FloatUtils](#)

Provides utility methods for compressing and decompressing float values.
 - class [FusionGlobalScriptableObject](#)

A base class for [ScriptableObjects](#) that are meant to be globally accessible, at edit-time and runtime. The way such objects are loaded is driven by usages of [FusionGlobalScriptableObjectSourceAttribute](#) attributes.
 - class [FusionGlobalScriptableObjectAttribute](#)

Provides additional information for a global scriptable object.
 - struct [FusionGlobalScriptableObjectLoadResult](#)

The result of [FusionGlobalScriptableObjectSourceAttribute.Load](#). Contains the loaded object and an optional un-loader delegate.
 - class [FusionGlobalScriptableObjectSourceAttribute](#)

Base class for all attributes that can be used to load [FusionGlobalScriptableObject](#). Attributes need to be registered at the assembly level. For instance, this snippet is used to register a default loader, that attempts to load from Resources based on [FusionGlobalScriptableObjectAttribute.DefaultPath](#):
 - class [FusionMonoBehaviour](#)

Base class for all [Fusion MonoBehaviour](#)s.
 - class [FusionScriptableObject](#)

Base class for all [Fusion scriptable objects](#).
 - class [HeapConfiguration](#)

Memory Heap Settings
 - class [HideArrayElementLabelAttribute](#)

Attribute used to hide the label of an array element in the inspector.
 - class [Hitbox](#)

Represents a single lag-compensated collider. Multiple component instances can be added anywhere in the hierarchy of a [NetworkObject](#) which includes a [HitboxRoot](#).
 - class [HitboxManager](#)

Entry point for lag compensated [Hitbox](#) queries, which maintains a history buffer, and provides lag compensated raycast and overlap methods. Singleton instance is accessible through the property [Runner.LagCompensation](#).
 - class [HitboxRoot](#)

Root [Hitbox](#) group container. Manages registering/unregistering hitboxes with the group, and defines the broadphase geometry for the group.
 - class [HostMigrationConfig](#)

Project configuration settings specific to how the Host Migration behaves.
 - class [HostMigrationToken](#)

Transitory Holder with all necessary information to restart the [Fusion Runner](#) after the Host Migration has completed
 - interface [IAfterAllTicks](#)

Interface for [AfterAllTicks](#) callback. Called after the re-simulation loop (when applicable), and also after the forward simulation loop. Implement this interface on [SimulationBehaviour](#) and [NetworkBehaviour](#) classes.
 - interface [IAfterClientPredictionReset](#)

Callback interface for [AfterClientPredictionReset](#). Called at the very start of the resimulation loop (on clients with prediction enabled), immediately after state is set to the latest server snapshot. Implement this interface on [SimulationBehaviour](#) and [NetworkBehaviour](#) classes.
 - interface [IAfterHostMigration](#)

Used to mark [NetworkBehaviors](#) that need to be react after a Host Migration process

- interface [IAfterRender](#)

Interface for [AfterRender](#) callback. Called after the render loop.
- interface [IAfterSpawned](#)

Interface for [AfterSpawned](#) callback. Called after the object is spawned.
- interface [IAfterTick](#)

Interface for [AfterTick](#) callback. Called after each tick simulation completes. Implement this interface on [SimulationBehaviour](#) and [NetworkBehaviour](#) classes.
- interface [IAfterUpdate](#)

Interface for the [AfterUpdate](#) callback, which is called at the end of each [Fusion](#) Update segment. Implement this interface on [SimulationBehaviour](#) and [NetworkBehaviour](#) classes.
- interface [IAfterUpdateRemotePrefabs](#)

Invoked after updating remote prefabs
- interface [IAsyncOperation](#)

Defines an asynchronous operation.
- interface [IBeforeAllTicks](#)

Interface for [BeforeAllTicks](#) callback. Called before the re-simulation loop (when applicable), and also before the forward simulation loop. Implement this interface on [SimulationBehaviour](#) and [NetworkBehaviour](#) classes.
- interface [IBeforeClientPredictionReset](#)

Callback interface for [BeforeClientPredictionReset](#). Called at the very start of the re-simulation loop (on clients with prediction enabled), before state is set to the latest server snapshot. Implement this interface on [SimulationBehaviour](#) and [NetworkBehaviour](#) classes.
- interface [IBeforeCopyPreviousState](#)

Interface for [BeforeCopyPreviousState](#) callback. Called before the copy of the previous state.
- interface [IBeforeHitboxRegistration](#)

Interface for [BeforeHitboxRegistration](#) callback. Implement this interface on [SimulationBehaviour](#) and [NetworkBehaviour](#) classes.
- interface [IBeforeSimulation](#)

Interface for [BeforeSimulation](#) callback. Called before both the re-simulation (when applicable) and forward simulation loops. Implement this interface on [SimulationBehaviour](#) and [NetworkBehaviour](#) classes.
- interface [IBeforeTick](#)

Interface for [BeforeTick](#) callback. Called before each tick is simulated. Implement this interface on [SimulationBehaviour](#) and [NetworkBehaviour](#) classes.
- interface [IBeforeUpdate](#)

Interface for the [BeforeUpdate](#) callback, which is called at the beginning of each [Fusion](#) Update segment. Implement this interface on [SimulationBehaviour](#) and [NetworkBehaviour](#) classes.
- interface [IBeforeUpdateRemotePrefabs](#)

Invoked before updating remote prefabs
- interface [ICoroutine](#)

Defines a coroutine.
- interface [IDespawned](#)

Interface for [Despawned](#) callback. Called when a [NetworkBehaviour](#) is despawned.
- interface [IElementReaderWriter](#)

Defines the interface for reading and writing elements in a byte array.
- interface [IFixedStorage](#)

Interface for fixed storage types.
- interface [IInputAuthorityGained](#)

Interface for handling the event when the input authority is gained.
- interface [IInputAuthorityLost](#)

Interface for handling the event when the input authority is lost.
- interface [IInterestEnter](#)

Interface for handling the event when a player enters the area of interest.
- interface [IInterestExit](#)

- Interface for handling the event when a player exits the area of interest.*

 - interface [ILocalPrefabCreated](#)

Interface for handling the event when a local prefab is created.
 - interface [INetworkArray](#)

Defines the interface for a networked array.
 - interface [INetworkAssetSource](#)

Interface for a network asset source.
 - interface [INetworkDictionary](#)

Defines the interface for a networked dictionary.
 - interface [INetworkInput](#)

Flag interface for custom [NetworkInput](#) structs.
 - interface [INetworkLinkedList](#)

Defines the interface for a networked linked list.
 - interface [INetworkObjectInitializer](#)

Interface for initializing network objects.
 - interface [INetworkObjectProvider](#)

Interface which defines the handlers for [NetworkRunner](#) [Spawn\(\)](#) and [Despawn\(\)](#) actions. Passing an instance of this interface to [NetworkRunner.StartGame\(StartGameArgs\)](#) as the [StartGameArgs.ObjectProvider](#) argument value will assign that instance as the handler for runner [Spawn\(\)](#) and [Despawn\(\)](#) actions. By default (if [StartGameArgs.ObjectProvider](#) == null) actions will use [Instantiate\(\)](#), and [Despawn\(\)](#) actions will use [Destroy\(\)](#).
 - interface [INetworkPrefabSource](#)

Interface for a network prefab source.
 - interface [INetworkRunnerCallbacks](#)

Interface for [NetworkRunner](#) callbacks. Register a class/struct instance which implements this interface with [NetworkRunner.AddCallbacks\(INetworkRunnerCallbacks\[\]\)](#).
 - interface [INetworkRunnerUpdater](#)

Interface which defines the handlers for [NetworkRunner](#) Updates. An implementation is responsible for calling [NetworkRunner.UpdateInternal\(double\)](#) and [NetworkRunner.RenderInternal](#) periodically.
 - interface [INetworkSceneManager](#)

Interface for a [NetworkRunner](#) scene manager. A scene manager is responsible for loading and unloading scenes
 - interface [INetworkStruct](#)

Base interface for all [Fusion](#) Network Structs
 - interface [INetworkTRSPTeleport](#)

Implement this interface on a [NetworkTRSP](#) implementation to indicate it can be teleported.
 - class [InlineHelpAttribute](#)

If applied to a field, checks if there is a help text for the field or the field's type and shows it in the inspector.
 - class [InterpolatedErrorCorrectionSettings](#)

A set of parameters that tune the interpolated correction of prediction error on transform data.
 - interface [IPlayerJoined](#)

Interface for handling the event when a player joins the game.
 - interface [IPlayerLeft](#)

Interface for handling the event when a player leaves the game.
 - interface [IRemotePrefabCreated](#)

Interface for handling the event when a remote prefab is created.
 - interface [ISceneLoadDone](#)

Interface for handling the event when a scene load operation is completed.
 - interface [ISceneLoadStart](#)

Interface for handling the event when a scene load operation is started.
 - interface [ISimulationEnter](#)

Interface for [SimulationEnter](#) callback. Called when the [NetworkObject](#) joins [AreaOfInterest](#). Implement this interface on [SimulationBehaviour](#) and [NetworkBehaviour](#) classes.
 - interface [ISimulationExit](#)

Interface for the [SimulationExit](#) callback. Called when the [NetworkObject](#) leaves AreaOfInterest. Implement this interface on [SimulationBehaviour](#) and [NetworkBehaviour](#) classes.

- interface [ISpawned](#)
 - Interface for handling the event when an object is spawned.
- interface [IStateAuthorityChanged](#)
 - Interface for handling the event when the state authority changes.
- struct [LagCompensatedHit](#)
 - Defines a lag compensated query hit result.
- class [LagCompensationSettings](#)
 - Settings for lag compensation history.
- class [LayerAttribute](#)
 - Specifies that an int field should be drawn as a layer field in the inspector.
- class [LayerMatrixAttribute](#)
 - Specifies that the integer array field should be drawn as a layer matrix in the inspector.
- class [LobbyInfo](#)
 - Holds information about a Lobby
- struct [Mask256](#)
 - [Mask256](#) is a 256-bit mask that can be used to store 256 boolean values.
- class [Maths](#)
 - Math utility methods.
- class [MaxStringByteCountAttribute](#)
 - Specifies that the string field should be drawn as a text field with a maximum byte count for given encoding.
- class [Native](#)
 - [Native Memory Allocator](#)
- class [NestedComponentUtilities](#)
 - Tools to replace GetComponent variants that respects nested objects. These are used to find components of a NetworkedObjects without also finding components that belong to parent or child NetworkedObjects.
- struct [NetworkArray](#)
 - [Fusion](#) type for networking arrays. Maximum capacity is fixed, and is set with the [CapacityAttribute](#).

- class [NetworkArrayExtensions](#)
 - Provides extension methods for the [NetworkArray](#) class.
- struct [NetworkArrayReadOnly](#)
 - Provides a read-only view of a network array.
- class [NetworkAssemblyIgnoreAttribute](#)
 - Network Assembly Ignore Attribute
- class [NetworkAssemblyWeavedAttribute](#)
 - Network Assembly Weaved Attribute
- class [NetworkBehaviour](#)
 - Base class for [Fusion](#) network components, which are associated with a [NetworkObject](#).
- struct [NetworkBehaviourBuffer](#)
 - Provides low level accesss to data buffers that can be read using a [NetworkBehaviourReader](#)
- struct [NetworkBehaviourBufferInterpolator](#)
 - The [NetworkBehaviourBufferInterpolator](#) struct is used to interpolate between two [NetworkBehaviourBuffer](#) instances. This is a read-only, ref struct, meaning it cannot be boxed and it can only be used on the stack.
- struct [NetworkBehaviourId](#)
 - Represents the unique identifier for a [NetworkBehaviour](#) instance.
- class [NetworkBehaviourUtils](#)
 - This static class provides utility methods for working with [NetworkBehaviour](#) objects.
- class [NetworkBehaviourWeavedAttribute](#)
 - Network [Behaviour](#) Weaved Attribute

- struct [NetworkBool](#)
Represents a boolean value that can be networked.
- struct [NetworkButtons](#)
Represents a set of buttons that can be networked.
- class [NetworkConfiguration](#)
Main network configuration class.
- class [NetworkDelegates](#)
Network Runner Callbacks Delegates
- class [NetworkDeserializeMethodAttribute](#)
Network Deserialize Method Attribute
- struct [NetworkDictionary](#)
Fusion type for networking Dictionaries. Maximum capacity is fixed, and is set with the [CapacityAttribute](#).

- struct [NetworkDictionaryReadOnly](#)
A read-only version of `NetworkDictionary< TKey, TValue >`.
- class [NetworkedAttribute](#)
- class [NetworkedWeavedArrayAttribute](#)
Attribute applied to an array property by the weaver.
- class [NetworkedWeavedAttribute](#)
Networked Weaved Attribute
- class [NetworkedWeavedDictionaryAttribute](#)
Attribute applied to a dictionary property by the weaver.
- class [NetworkedWeavedLinkedListAttribute](#)
Attribute applied to a list property by the weaver.
- class [NetworkEvents](#)
Companion component for [NetworkRunner](#). Exposes `INetworkRunnerCallbacks` as `UnityEvents`, which can be wired up to other components in the inspector.
- struct [NetworkId](#)
The unique identifier for a network entity.
- struct [NetworkInput](#)
NetworkInput Struct
- class [NetworkInputUtils](#)
Utility methods for [NetworkInput](#)
- class [NetworkInputWeavedAttribute](#)
Network Input Weaved Attribute
- struct [NetworkLinkedList](#)
Fusion type for networking LinkedLists. Maximum capacity is fixed, and is set with the [CapacityAttribute](#).

Typical Usage:
- struct [NetworkLinkedListReadOnly](#)
Read-only version of `NetworkLinkedList< T >`.
- struct [NetworkLoadSceneParameters](#)
Parameters for loading a scene
- class [NetworkMecanimAnimator](#)
A component for synchronizing the Animator controller state from the State Authority to network proxies. Requires a Unity Animator component, and a [NetworkObject](#) component. NOTE: Animator Root Motion is not compatible with re-simulation and prediction.
- class [NetworkObject](#)
The primary [Fusion](#) component for networked GameObject entities. This stores the object's network identity and manages the object's state and input authority.
- class [NetworkObjectFlagsExtensions](#)

- Extension methods for the NetworkObjectFlags enum.*
- struct [NetworkObjectGuid](#)
NetworkObjectGuid
 - struct [NetworkObjectHeader](#)
Network object header information for a [NetworkObject](#).
 - struct [NetworkObjectHeaderPtr](#)
Represents a pointer to a [NetworkObjectHeader](#). This struct is unsafe because it uses pointers.
 - class [NetworkObjectInitializerUnity](#)
Initializes network objects for Unity.
 - class [NetworkObjectMeta](#)
Meta information about a network object.
 - struct [NetworkObjectNestingKey](#)
A key used to identify a network object nesting.
 - class [NetworkObjectPrefabData](#)
This class represents the data for a network object prefab.
 - class [NetworkObjectProviderDummy](#)
A dummy implementation of the [INetworkObjectProvider](#) interface. This class is used for testing purposes and throws a [NotImplementedException](#) for all its methods.
 - struct [NetworkObjectReleaseContext](#)
Represents the context for releasing a network object. This struct is unsafe because it uses pointers.
 - class [NetworkObjectSortKeyComparer](#)
This class is used to compare two [NetworkObject](#) instances based on their SortKey. It implements the [IComparer](#) interface.
 - class [NetworkObjectSpawnException](#)
Network Object Spawn Exception
 - struct [NetworkObjectTypeId](#)
ID for a [NetworkObject](#) Prefab which has been cataloged in a [NetworkProjectConfig.PrefabTable](#).
 - struct [NetworkPhysicsInfo](#)
Network Physics [INetworkStruct](#)
 - struct [NetworkPrefabAcquireContext](#)
Represents the context for acquiring a prefab instance for a network object. This struct is unsafe because it uses pointers.
 - class [NetworkPrefabAttribute](#)
Network Prefab Attribute
 - struct [NetworkPrefabId](#)
ID for a [NetworkObject](#) Prefab which has been cataloged in a [NetworkProjectConfig.PrefabTable](#).
 - struct [NetworkPrefabInfo](#)
Meta data for a [NetworkObject](#) prefab which has been cataloged in a [NetworkProjectConfig.PrefabTable](#).
 - struct [NetworkPrefabRef](#)
NetworkPrefabRef
 - class [NetworkPrefabTable](#)
Class representing a table of network prefabs.
 - struct [NetworkPrefabTableOptions](#)
Options for the [NetworkPrefabTable](#).
 - class [NetworkProjectConfig](#)
The core [Fusion](#) config file that is shared with all peers at startup.
 - class [NetworkProjectConfigAsset](#)
Manages and references the current instance of [NetworkProjectConfig](#)
 - struct [NetworkRNG](#)
PCG32 random generator, 16 bytes in size. <http://www.pcg-random.org>
 - class [NetworkRpcStaticWeavedInvokerAttribute](#)

- Network Rpc Static Weaved Invoker Attribute Contains info about a static weaved RPC Method*

 - class [NetworkRpcWeavedInvokerAttribute](#)
- Network Rpc Weaved Invoker Attribute Contains info about a weaved RPC Method*

 - class [NetworkRunner](#)
- Host Migration related code in order to get a copy of the [Simulation](#) State*

 - class [NetworkRunnerCallbackArgs](#)
- Stores data types used on the [INetworkRunnerCallbacks](#) interface*

 - class [NetworkRunnerUpdaterDefault](#)
- Default implementation of [INetworkRunnerUpdater](#) that uses the Unity PlayerLoop.*

 - struct [NetworkRunnerUpdaterDefaultInvokeSettings](#)
- Settings for the [NetworkRunnerUpdaterDefault](#).*

 - struct [NetworkSceneAsyncOp](#)
- A wrapper for async scene operations.*

 - struct [NetworkSceneInfo](#)
- Can store up to 8 active scenes and allows for duplicates. Each write increases [Version](#) which can be used to generate unique scene objects ids for when a scene is supposed to be reloaded.*

 - struct [NetworkSceneLoadId](#)
- A unique identifier for a scene load operation.*

 - struct [NetworkSceneObjectId](#)
- A unique identifier for a scene object.*

 - class [NetworkSerializeMethodAttribute](#)
- Network Serialize Method Attribute*

 - class [NetworkSimulationConfiguration](#)
- Configuration for network conditions simulation (induced latency and loss).*

 - struct [NetworkSpawnOp](#)
- Spawn Operation*

 - class [NetworkString](#)
- Fixed-size UTF32 string. All operations are alloc-free, except for converting to [System.String](#).*

 - class [NetworkStructUtils](#)
- Utility methods for [INetworkStruct](#)*

 - class [NetworkStructWeavedAttribute](#)
- Describes the total number of WORDs a [INetworkStruct](#) uses.*

 - class [NetworkTransform](#)
- Add to any [NetworkObject](#) Transform, or its associated child Transforms to automatically synchronize TRSP (Position/Rotation/Scale/Parent).*

 - class [NetworkTRSP](#)
- Base class for spatial (Position/Rotation/Scale/Parent) synchronization component, such as [NetworkTransform](#). Provides the base logic for render interpolation, parenting synchronization, and teleport, that can be used in components derived from this class.*

 - struct [NetworkTRSPData](#)
- Data structure storing spatial (Position/Rotation/Scale/Parent) synchronization data for spatial synchronization components, [NetworkTRSP](#) and its subclass [NetworkTransform](#).*

 - class [NormalizedRectAttribute](#)
- Enables a special inspector drawer for Unity Rect type, specially designed for editing RectTransforms using normalized values.*

 - class [OnChangedRenderAttribute](#)
- OnChangedRender Attribute*

 - struct [PlayerRef](#)
- Represents a [Fusion](#) player.*

 - class [PreserveInPluginAttribute](#)
- Preserve In Plugin Attribute*

 - class [Primes](#)

- Provides a set of methods to work with prime numbers.*

 - class [PropertyAttribute](#)
 - Specifies that the attribute can be applied to fields only.*
 - struct [Ptr](#)
 - Ptr*
 - struct [QuaternionCompressed](#)
 - Represents a compressed Quaternion value for network transmission.*
 - class [RangeExAttribute](#)
 - Represents an attribute that specifies a range of values for a field or property.*
 - class [ReadOnlyAttribute](#)
 - Attribute used to mark a field as read-only.*
 - class [ReadWriteUtils](#)
 - Provides utility methods for reading and writing data.*
 - class [ReadWriteUtilsForWeaver](#)
 - Provides utility methods for reading and writing data.*
 - class [ReflectionUtils](#)
 - Provides utility methods for reflection.*
 - class [RenderAttribute](#)
 - Override default render settings for [Networked] properties.*
 - struct [RenderTimeline](#)
 - Can be used to acquire interpolated data for different points in time.*
 - class [RenderWeavedAttribute](#)
 - Render Weaved Attribute*
 - class [ResolveNetworkPrefabSourceAttribute](#)
 - Resolve Network Prefab Source Attribute*
 - class [RpcAttribute](#)
 - Flags a method as being a networked Remote Procedure Call. Only usable in a [NetworkBehaviour](#). Calls to this method (from the indicated allowed [RpcSources](#)) will generate a network message, which will execute the method remotely on the indicated [RpcTargets](#). The RPC method can include an empty [RpcInfo](#) argument, that will include meta information about the RPC on the receiving peer.*
 - struct [RpcHeader](#)
 - Header for RPC messages.*
 - struct [RpcInfo](#)
 - [RpcInfo](#) is a struct that contains information about the RPC message.*
 - struct [RpcInvokeData](#)
 - Represents the data required to invoke an RPC message.*
 - struct [RpcInvokeInfo](#)
 - May be used as an optional [RpcAttribute](#) return value. Contains meta data about the RPC send, such as failure to send reasons, culling, message size, etc.*
 - struct [RpcSendResult](#)
 - RPC send operation result information.*
 - class [RpcTargetAttribute](#)
 - RPC attribute used to indicate a specific target player for an RPC when sending from one player to another. RPC is sent to the server, and then is forwarded to the specified player. Usage:*
 - class [RuntimeUnityFlagsSetup](#)
 - Contains methods to setup runtime flags for Unity.*
 - struct [SceneLoadDoneArgs](#)
 - Struct that contains information about a scene after it has been loaded.*
 - class [ScenePathAttribute](#)
 - Specifies that a string field represents a scene path.*
 - struct [SceneRef](#)
 - Scene reference struct. Can be used to reference a scene by index or by path.*

- class [ScriptHelpAttribute](#)
Defines the appearance of the script header in the Unity inspector.
- class [SerializableDictionary](#)
A serializable dictionary.
- struct [SerializableType](#)
A System.Type wrapper that can be serialized.
- class [SerializableTypeAttribute](#)
Specifies that either a string field represents a type name or sets additional options for [SerializableType](#) field.
- class [SerializeReferenceTypePickerAttribute](#)
Attribute used to show a type picker for a field with [\[SerializeReference\]](#).
- class [SessionInfo](#)
Holds information about the Game Session
- class [Simulation](#)
Main simulation class
- class [SimulationBehaviour](#)
Base class for a [Fusion](#) aware [Behaviour](#) (derived from [UnityEngine.MonoBehaviour](#)). If a [SimulationBehaviour](#) is found on a [NetworkRunner](#) game object during the runner initialisation, the [SimulationBehaviour](#) is automatically registered. Objects derived from this object can be associated with a [NetworkRunner](#) and [Simulation](#) using [NetworkRunner.AddGlobal\(\)](#).
- class [SimulationBehaviourAttribute](#)
Attribute for specifying which [SimulationStages](#) and [SimulationModes](#) this [SimulationBehaviour](#) will execute in. Can be used to limit execution to only Host, Server or Client peers, or to only execute on Resimulation or Forward ticks. Usage:
- struct [SimulationBehaviourListScope](#)
Provides a scope for a [SimulationBehaviourUpdater.BehaviourList](#), incrementing its lock count on creation and decrementing it on disposal. If the lock count reaches zero on disposal, all pending removals in the list are performed.
- class [SimulationConfig](#)
Project configuration settings specific to how the [Simulation](#) class behaves.
- class [SimulationInput](#)
Simulation Input
- struct [SimulationInputHeader](#)
Simulation Input Header
- struct [SimulationMessage](#)
Simulation Message
- struct [SimulationMessagePtr](#)
Simulation Message Pointer
- struct [SimulationRuntimeConfig](#)
Stores the runtime configuration of the simulation
- struct [StartGameArgs](#)
[Fusion](#) Start Arguments, used to configure the simulation mode and other settings
- class [StartGameResult](#)
Represents the result of starting the [Fusion Simulation](#)
- struct [Tick](#)
A tick is a 32-bit integer that represents a frame number.
- struct [TickAccumulator](#)
A tick accumulator.
- struct [TickRate](#)
A tick rate is a collection of tick rates.
- struct [TickTimer](#)
A timer that is based on ticks instead of seconds.
- struct [Timer](#)
Represents a high-resolution timer.

- class [TimeSyncConfiguration](#)
Time Synchronization Configuration
- class [ToggleLeftAttribute](#)
Specifies that the bool field should be drawn as the toggle on the left side of the label.
- class [UnityAttribute](#)
Unity Attribute class. Used to mark a field with the respective [Units](#)
- class [UnityAddressablesRuntimeKeyAttribute](#)
Specifies that the string field represents a key for Unity Addressables.
- class [UnityAssetGuidAttribute](#)
Specifies that the string field represents a GUID of an asset.
- class [UnityContextMenuMenuItemAttribute](#)
Unity ContextMenuItemAttribute
- class [UnityDelayedAttribute](#)
Unity DelayedAttribute
- class [UnityFormerlySerializedAsAttribute](#)
Unity FormerlySerializedAsAttribute
- class [UnityHeaderAttribute](#)
Unity HeaderAttribute
- class [UnityMinAttribute](#)
Unity MinAttribute
- class [UnityMultilineAttribute](#)
Unity MultilineAttribute
- class [UnityNonReorderableAttribute](#)
Unity NonReorderableAttribute
- class [UnityNonSerializedAttribute](#)
Unity NonSerializedAttribute
- class [UnityRangeAttribute](#)
Unity RangeAttribute
- class [UnityResourcePathAttribute](#)
Specifies that the string field represents a path to a Unity resource.
- class [UnitySerializeField](#)
Unity SerializeField
- class [UnitySerializeReference](#)
Unity SerializeReference
- class [UnitySpaceAttribute](#)
Unity SpaceAttribute
- class [UnityTooltipAttribute](#)
Unity TooltipAttribute
- class [UTF32Tools](#)
[UTF32Tools](#) provides a set of methods to work with UTF32 encoded strings.
- struct [Vector2Compressed](#)
Represents a compressed Vector2 value for network transmission.
- struct [Vector3Compressed](#)
Represents a compressed Vector3 value for network transmission.
- struct [Vector4Compressed](#)
Represents a compressed Vector4 value for network transmission.
- class [Versioning](#)
The [Versioning](#) class provides methods and properties related to versioning.
- class [WarnIfAttribute](#)
Editor attribute for adding notices to fields if the condition member evaluates as `true`. Condition member can be a property, field or method (with a return value).
- class [WeaverGeneratedAttribute](#)
Weaver Generated Attribute

Enumerations

- enum class **AnimatorSyncSettings**
- enum class [CompareOperator](#)
Comparison method for evaluating condition member value against compareToValues.
- enum class [ConnectionType](#)
Defines the type of the current connection with the Remote Peer, either the Server or a Client
- enum class [DrawIfMode](#)
Mode for the DrawIf attribute. If the condition is not met, should the field be hidden or just read-only?
- enum class [EditorButtonVisibility](#)
Specifies the visibility options for an editor button.
- enum class [GameMode](#)
Fusion Game Mode.
- enum class [HitboxTypes](#)
Defines the collision geometry type of a Hitbox.
- enum class [HitOptions](#)
- enum class [NetworkObjectAcquireResult](#)
Enum representing the possible results of acquiring a prefab instance for a network object.
- enum class **NetworkObjectConnectionDataStatus**
- enum class **NetworkObjectDestroyFlags**
- enum class [NetworkObjectFlags](#) : int
Enum representing the flags for network objects in the Fusion system. This enum is marked with the Flags attribute, allowing it to be treated as a bit field.
- enum class [NetworkObjectHeaderFlags](#) : int
Enum representing various flags for a network object header. Each flag represents a different state or property of a network object.
- enum class **NetworkObjectHeaderPlayerDataFlags** : int
- enum class **NetworkObjectPacketFlags**
- enum class **NetworkObjectRuntimeFlags** : int
- enum class [NetworkPrefabTableGetPrefabResult](#)
Enum representing the possible results of attempting to get a prefab from the NetworkPrefabTable.
- enum class [NetworkSceneInfoChangeSource](#)
What has contributed to the observed change in the scene info.
- enum class [NetworkSceneInfoDefaultFlags](#) : uint
Network Scene Info Default Flags
- enum class [NetworkSpawnFlags](#) : short
Network Spawn Flags
- enum class [NetworkSpawnStatus](#) : int
Network Spawn Status
- enum class [NetworkTypeIdKind](#)
Enum representing the type of a NetworkObject.
- enum class [PageSizes](#)
Page Bit Shift Lookup Table
- enum class [PriorityLevel](#)
Enum representing the priority levels for network objects
- enum class [RenderSource](#)
Indicates how available snapshot data should be used to render networked properties (in the chosen RenderTimeframe).
- enum class [RenderTimeframe](#)
Indicates which point in time (or "timeframe") networked properties should be rendered in.
- enum class [RpcChannel](#)
Flags for the RPC channel.

- enum class [RpcHostMode](#)
Options for when the game is run in SimulationModes.Host mode and RPC is invoked by the host.
- enum class [RpcLocalInvokeResult](#)
Results for the local RPC Invocation of the RPC method.
- enum class [RpcSendCullResult](#)
Results for the RPC message send operation. Note: Some individual targets may be culled even if the send operation succeeds. Information about culled targets can be found in RpcInvokeInfo.SendResult.
- enum class [RpcSendMessageResult](#)
Result flags for the RPC message send operation.
- enum class [RpcSources](#)
Enum representing the sources of an RPC message.
- enum class [RpcTargets](#)
Enum representing the targets of an RPC message.
- enum class [RpcTargetStatus](#)
Enum representing the status of an RPC target.
- enum class [ScriptHeaderBackColor](#)
Color of the component graphic header in the Unity inspector. None indicates no header graphic should be used.
- enum class [ScriptHeaderIcon](#)
Icon to be rendered on the component graphic header in the Unity inspector.
- enum class [ScriptHeaderStyle](#)
Style of the script header in the Unity inspector.
- enum class [SessionLobby](#)
Session Lobby Type
- enum class [ShutdownReason](#)
Describes a list of Reason why the Fusion Runner was Shutdown
- enum class **SimulationBehaviourRuntimeFlags**
- enum class **SimulationMessageInternalTypes**
- enum class [SimulationModes](#)
Flags for The type of network peer a simulation represents.
- enum class [SimulationStages](#)
Flags for which stage the simulation currently running. Forward is when a tick is being simulated for the first time. Resimulate is when a tick is being simulated again with corrections.
- enum class [Topologies](#)
- enum class [Units](#)
Unit Type for a certain field. This helps to identify the unit that a certain value represents, like Seconds or Percentage
- enum class [UnityPlayerLoopSystemAddMode](#)
Enum representing the possible modes for adding a system to the Unity player loop.

Functions

- delegate void [FusionGlobalScriptableObjectUnloadDelegate](#) ([FusionGlobalScriptableObject](#) instance)
A delegate that can be used to unload a FusionGlobalScriptableObject.
- delegate void [NetworkObjectSpawnDelegate](#) ([NetworkSpawnOp](#) result)
Network Object Spawn Delegate
- unsafe delegate void [RpcInvokeDelegate](#) ([NetworkBehaviour](#) behaviour, [SimulationMessage](#) *message)
Represents a delegate that can be invoked by an RPC message.
- unsafe delegate void [RpcStaticInvokeDelegate](#) ([NetworkRunner](#) runner, [SimulationMessage](#) *message)
Represents a delegate that can be invoked by an RPC message.

5.1.1 Enumeration Type Documentation

5.1.1.1 CompareOperator

```
enum CompareOperator [strong]
```

Comparison method for evaluating condition member value against compareToValues.

Enumerator

Equal	true if condition member value equals compareToValue.
NotEqual	true if condition member value is not equal to compareToValue.
Less	true if condition member value is less than compareToValue.
LessOrEqual	true if condition member value is less than or equal to compareToValue.
GreaterOrEqual	true if condition member value is greater than or equal to compareToValue.
Greater	true if condition member value is greater than compareToValue.
NotZero	Returns true if the condition member evaluates to anything other than zero. In the case of object references, this means true for any non-null value.
IsZero	Returns true if the condition member evaluates to zero. In the case of object references, this means true for any null value.
BitwiseAndNotEqualZero	Returns true if the bitwise AND of the condition member and compareToValue is not zero.

5.1.1.2 ConnectionType

```
enum ConnectionType [strong]
```

Defines the type of the current connection with the Remote Peer, either the Server or a Client

Enumerator

None	No connection is currently active
Relayed	Connection was accomplished using the Photon Relay Services
Direct	Connection was accomplished directly with the remote peer

5.1.1.3 DrawIfMode

```
enum DrawIfMode [strong]
```

Mode for the DrawIf attribute. If the condition is not met, should the field be hidden or just read-only?

Enumerator

ReadOnly	Field is read-only if the condition is not met.
Hide	Field is hidden if the condition is not met.

5.1.1.4 EditorButtonVisibility

```
enum EditorButtonVisibility [strong]
```

Specifies the visibility options for an editor button.

Enumerator

PlayMode	The button is only visible in Play Mode.
EditMode	The button is only visible in Edit Mode.
Always	The button is always visible.

5.1.1.5 GameMode

```
enum GameMode [strong]
```

[Fusion](#) Game Mode.

Used to select how the local simulation will act.

Enumerator

Single	Single Player Mode: it works very similar to Host Mode, but don't accept any connections.
Shared	Shared Mode: starts a Game Client, which will connect to a Game Server running in the Photon Cloud using the Fusion Plugin.
Server	Server Mode: starts a Dedicated Game Server with no local player.
Host	Host Mode: starts a Game Server and allows a local player.
Client	Client Mode: starts a Game Client, which will connect to a peer in either Server or Host Modes.
AutoHostOrClient	Automatically start as Host or Client. The first peer to connect to a room will be started as a Host, all others will connect as clients.

5.1.1.6 HitboxTypes

```
enum HitboxTypes [strong]
```

Defines the collision geometry type of a [Hitbox](#).

Enumerator

None	[Future Use] to represent a disabled Hitbox .
Box	Geometry is a box, fill in Extents and (optional) Offset.
Sphere	Geometry is a sphere, fill in Radius and (optional) Offset.
Capsule	Geometry is a capsule, fill in capsule Radius, capsule Height and (optional) Offset.

5.1.1.7 HitOptions

```
enum HitOptions [strong]
```

Per-query options for lag compensation (both raycast and overlap).

Enumerator

None	Default, no extra options.
IncludePhysX	Add this to include checks against PhysX colliders.
IncludeBox2D	Add this to include checks against Box2D colliders. If PhysX flag is set, it will be used instead.
SubtickAccuracy	Subtick accuracy query (exactly like seen by player).
IgnoreInputAuthority	If the HitboxRoot objects which the player performing the query (if specified) has input authority over should be ignored by the query.

5.1.1.8 NetworkObjectAcquireResult

```
enum NetworkObjectAcquireResult [strong]
```

Enum representing the possible results of acquiring a prefab instance for a network object.

Enumerator

Success	Indicates that the prefab instance was successfully acquired.
Failed	Indicates that the acquisition of the prefab instance failed.
Retry	Indicates that the acquisition of the prefab instance should be retried.
Ignore	Indicates that the acquisition of the prefab instance should be ignored.

5.1.1.9 NetworkObjectFlags

```
enum NetworkObjectFlags : int [strong]
```

Enum representing the flags for network objects in the [Fusion](#) system. This enum is marked with the Flags attribute, allowing it to be treated as a bit field.

Enumerator

None	Represents a state where no flags are set.
MaskVersion	Mask for isolating the version part of the flags.
V1	Represents the first version of the network object flags.
Ignore	Flag indicating that the network object should be ignored.
MasterClientObject	Shared Mode only. The current Master client always has State Authority for this object. If the Master client leaves the new Master client automatically becomes the State Authority for it.
DestroyWhenStateAuthorityLeaves	Flag indicating that the network object should be destroyed when the state authority leaves.
AllowStateAuthorityOverride	Flag indicating that the network object allows state authority override.

5.1.1.10 NetworkObjectHeaderFlags

```
enum NetworkObjectHeaderFlags : int [strong]
```

Enum representing various flags for a network object header. Each flag represents a different state or property of a network object.

Enumerator

GlobalObjectInterest	Flag indicating that the object is of global interest.
DestroyWhenStateAuthorityLeaves	Flag indicating that the object should be destroyed when the state authority leaves.
SpawnedByClient	Flag indicating that the object was spawned by a client.
AllowStateAuthorityOverride	Flag indicating that the state authority override is allowed.
Struct	Flag indicating that the object is a struct.
StructArray	Flag indicating that the object is an array of structs.
DontDestroyOnLoad	Flag indicating that the object should not be destroyed on load.
HasMainNetworkTRSP	Flag indicating that the object has a main network TRSP.
AreaOfInterest	Flag indicating that the object is in an area of interest.

5.1.1.11 NetworkPrefabTableGetPrefabResult

```
enum NetworkPrefabTableGetPrefabResult [strong]
```

Enum representing the possible results of attempting to get a prefab from the [NetworkPrefabTable](#).

Enumerator

Success	The prefab was successfully retrieved.
InProgress	The retrieval of the prefab is in progress.
NotFound	The prefab was not found in the NetworkPrefabTable .
LoadError	There was an error in loading the prefab.

5.1.1.12 NetworkSceneInfoChangeSource

```
enum NetworkSceneInfoChangeSource [strong]
```

What has contributed to the observed change in the scene info.

Enumerator

None	No change.
Initial	The initial local scene has changed.
Remote	The remove scene has changed.

5.1.1.13 NetworkSceneInfoDefaultFlags

```
enum NetworkSceneInfoDefaultFlags : uint [strong]
```

Network Scene Info Default Flags

Enumerator

SceneCountMask	The scene count mask
ConterMask	The counter mask

5.1.1.14 NetworkSpawnFlags

```
enum NetworkSpawnFlags : short [strong]
```

Network Spawn Flags

Enumerator

DontDestroyOnLoad	Object get spawned as DontDestroyOnLoad on all clients.
SharedModeStateAuthMasterClient	In shared mode, override the state authority to PlayerRef.MasterClient , ignoring "Is Master Client Object" inspector setting. If used by a non-master client, object will be spawned with local player authority and an error message will be logged.
SharedModeStateAuthLocalPlayer	In shared mode, override the state authority to local player, ignoring "Is Master Client Object" inspector setting.

5.1.1.15 NetworkSpawnStatus

```
enum NetworkSpawnStatus : int [strong]
```

Network Spawn Status

Enumerator

Queued	Spawn is queued and will be spawned when the prefab is loaded.
Spawned	Spawned successfully.
FailedToLoadPrefabSynchronously	Failed to Load Prefab Synchronously.
FailedToCreateInstance	Failed to create instance.
FailedClientCantSpawn	Failed to spawn because the client can't spawn.
FailedLocalPlayerNotYetSet	Failed to spawn because the local player is not yet set.

5.1.1.16 NetworkTypeIdKind

```
enum NetworkTypeIdKind [strong]
```

Enum representing the type of a [NetworkObject](#).

Enumerator

Prefab	Represents a NetworkObject that is a Prefab.
Custom	Represents a NetworkObject that is a Custom type.
InternalStruct	Represents a NetworkObject that is an InternalStruct.
SceneObject	Represents a NetworkObject that is a SceneObject.
Invalid	Represents an Invalid NetworkObject type.

5.1.1.17 PageSizes

```
enum PageSizes [strong]
```

Page Bit Shift Lookup Table

5.1.1.18 PriorityLevel

```
enum PriorityLevel [strong]
```

Enum representing the priority levels for network objects

Enumerator

Player	Priority level assigned to player-related network objects.
High	High priority level, typically assigned to network objects that require frequent updates.
Medium	Medium priority level, typically assigned to network objects that require regular updates.
Low	Low priority level, typically assigned to network objects that require less frequent updates.
Lowest	Lowest priority level, typically assigned to network objects that require minimal updates.

5.1.1.19 RenderSource

```
enum RenderSource [strong]
```

Indicates how available snapshot data should be used to render networked properties (in the chosen [RenderTimeframe](#)).

Enumerator

Interpolated	The rendered value will come from interpolating the values at From and To to the desired point in time.
From	The rendered value will come from the nearest available snapshot at or before the point in time being rendered.
To	The rendered value will come from the nearest available snapshot ahead of the point in time being rendered.
Latest	The rendered value will come from the latest snapshot.

5.1.1.20 RenderTimeframe

```
enum RenderTimeframe [strong]
```

Indicates which point in time (or "timeframe") networked properties should be rendered in.

Enumerator

Local	The default timeframe for owned and predicted objects.
Remote	The default timeframe for proxied objects.

5.1.1.21 RpcChannel

```
enum RpcChannel [strong]
```

Flags for the RPC channel.

Enumerator

Reliable	Rpc order preserved, delivery verified, resend in case of a failed delivery.
Unreliable	Rpc order preserved, delivery not verified, no resend attempts.

5.1.1.22 RpcHostMode

```
enum RpcHostMode [strong]
```

Options for when the game is run in [SimulationModes.Host](#) mode and RPC is invoked by the host.

Enumerator

SourcelsServer	If host invokes RPC RpcInfo.Source will be set to PlayerRef.None (default).
SourcelsHostPlayer	If host invokes RPC RpcInfo.Source will be set to the host's local player.

5.1.1.23 RpcLocalInvokeResult

```
enum RpcLocalInvokeResult [strong]
```

Results for the local RPC Invocation of the RPC method.

Enumerator

Invoked	RPC has been invoked locally.
NotInvokableLocally	Not invoked locally because RpcAttribute.InvokeLocal is false.
NotInvokableDuringResim	Not invoked locally because InvokeResim is false and simulation stage is SimulationStages.Resimulate
InsufficientSourceAuthority	Not invoked because source NetworkObject current authority does not match flags set in RpcAttribute.Sources
InsufficientTargetAuthority	Not invoked because target player is local and this NetworkObject current authority does not match flags set in RpcAttribute.Targets
TargetPlayerIsNotLocal	Not invoked because target player is not local.
PayloadSizeExceeded	RPC is too large. See RpcAttribute.MaxPayloadSize for the maximum allowed size.
TargetPlayerIsNotLocal	TargetPlayerIsNotLocal

5.1.1.24 RpcSendCullResult

```
enum RpcSendCullResult [strong]
```

Results for the RPC message send operation. Note: Some individual targets may be culled even if the send operation succeeds. Information about culled targets can be found in [RpclInvokeInfo.SendResult](#).

Enumerator

NotCulled	RPC has been sent. Check RpcInvokeInfo.SendResult for details.
NotInvokableDuringResim	Send culled because RpcAttribute.InvokeLocal is false.
InsufficientSourceAuthority	Send culled because source NetworkObject current authority does not match flags set in RpcAttribute.Sources
NoActiveConnections	Send culled because there are no active connections.
TargetPlayerUnreachable	Send culled because target player does not exist.
TargetPlayerIsLocalButRpcsNotInvokableLocally	Send culled because target player is local and RpcAttribute.InvokeLocal is false.
PayloadSizeExceeded	RPC message size is too large to be sent. See RpcAttribute.MaxPayloadSize for the maximum allowed size.

5.1.1.25 **RpcSendMessageResult**

```
enum RpcSendMessageResult [strong]
```

Result flags for the RPC message send operation.

Enumerator

None	Invalid result.
SentToServerForForwarding	Client sent to the server, server will send to the target client.
SentToTargetClient	Server sent to a specific client (a targeted message).
SentBroadcast	Server attempted to send to all the clients and at least one succeeded.
NotSentTargetObjectNotConfirmed	Target object not confirmed on the client.
NotSentTargetObjectNotInPlayerInterest	Target object not in client's interest. Likely due to being outside of player's AOI region, or needs to be explicitly set as always interested.
NotSentTargetClientNotAvailable	Target client not connected (a targeted message).
NotSentBroadcastNoActiveConnections	Server attempted to send to all the clients, but none was connected.
NotSentBroadcastNoConfirmedNorInterestedClients	Server attempted to send to all the clients, but the target object is not confirmed/not in Object Interest for all target clients.
MaskSent	Mask for sent messages.
MaskNotSent	Mask for not sent messages.
MaskBroadcast	Mask for broadcast messages.
MaskCulled	Mask for culled messages.

5.1.1.26 **RpcSources**

```
enum RpcSources [strong]
```

Enum representing the sources of an RPC message.

Enumerator

StateAuthority	Represents the state authority source of an RPC message.
InputAuthority	Represents the input authority source of an RPC message.
Proxies	Represents the proxy source of an RPC message.
All	Represents all possible sources of an RPC message.

5.1.1.27 RpcTargets

```
enum RpcTargets [strong]
```

Enum representing the targets of an RPC message.

Enumerator

StateAuthority	Represents the state authority target of an RPC message.
InputAuthority	Represents the input authority target of an RPC message.
Proxies	Represents the proxy target of an RPC message.
All	Represents all possible targets of an RPC message.

5.1.1.28 RpcTargetStatus

```
enum RpcTargetStatus [strong]
```

Enum representing the status of an RPC target.

Enumerator

Unreachable	Represents an unreachable RPC target.
Self	Represents the RPC target as self.
Remote	Represents a remote RPC target.

5.1.1.29 ScriptHeaderBackColor

```
enum ScriptHeaderBackColor [strong]
```

Color of the component graphic header in the Unity inspector. None indicates no header graphic should be used.

5.1.1.30 ScriptHeaderIcon

enum `ScriptHeaderIcon` [strong]

Icon to be rendered on the component graphic header in the Unity inspector.

5.1.1.31 ScriptHeaderStyle

enum `ScriptHeaderStyle` [strong]

Style of the script header in the Unity inspector.

Enumerator

Unity	Use the default Unity header style.
Photon	Use the Photon header style.

5.1.1.32 SessionLobby

enum `SessionLobby` [strong]

Session Lobby Type

Enumerator

Invalid	Invalid Session Lobby Type
ClientServer	ClientServer Lobby
Shared	Shared Lobby
Custom	Custom Lobby - works in conjunction with a Lobby Name/ID

5.1.1.33 ShutdownReason

enum `ShutdownReason` [strong]

Describes a list of Reason why the `Fusion` Runner was Shutdown

Enumerator

Ok	OK Reason means <code>Fusion</code> was Shutdown by request
Error	Shutdown was caused by some internal error
IncompatibleConfiguration	Raised when the peer tries to Join a Room with a mismatching type between ClientServer Mode and Shared Mode.

Enumerator

ServerInRoom	Raised when the local peer started as a Server and tried to join a Room that already has a Server peer.
DisconnectedByPluginLogic	Raised when the Peer is disconnected or kicked by a Plugin Logic.
GameClosed	Raised when the Game the Peer is trying to Join is Closed
GameNotFound	Raised when the Game the Peer is trying to Join does not exist
MaxCcuReached	Raised when all CCU available for the Photon Application are in use
InvalidRegion	Raised when the peer is trying to connect to an unavailable or non-existent Region
GameIdAlreadyExists	Raised when a Session with the same name was already created
GamelsFull	Raised when a peer is trying to join a Room with already the max capacity of players
InvalidAuthentication	Raised when the Authentication Values are invalid
CustomAuthenticationFailed	Raised when the Custom Authentication has failed for some other reason
AuthenticationTicketExpired	Raised when the Authentication Ticket has expired
PhotonCloudTimeout	Timeout on the Connection with the Photon Cloud
AlreadyRunning	Raised when Fusion is already running and the StartGame is invoked again
InvalidArguments	Raised when any of the StartGame arguments does not meet the requirements
HostMigration	Signal this Runner is shutting down because of a Host Migration is about to happen
ConnectionTimeout	Connection with a remote server failed by timeout
ConnectionRefused	Connection with a remote server failed because it was refused
OperationTimeout	The current operation has timed out
OperationCanceled	The current operation was canceled

5.1.1.34 SimulationModes

```
enum SimulationModes [strong]
```

Flags for The type of network peer a simulation represents.

Enumerator

Server	Simulation represents a server peer, with no local player.
Host	Simulation represents a server peer, with a local player.
Client	Simulation represents a client peer, with a local player.

5.1.1.35 SimulationStages

```
enum SimulationStages [strong]
```

Flags for which stage the simulation currently running. Forward is when a tick is being simulated for the first time. Resimulate is when a tick is being simulated again with corrections.

Enumerator

Forward	Currently simulating a tick for the first time.
Resimulate	Currently simulating a previously simulated tick again, with state corrections.

5.1.1.36 Topologies

```
enum Topologies [strong]
```

Enumerator

ClientServer	Classic server and client model
Shared	Relay based shared world model

5.1.1.37 Units

```
enum Units [strong]
```

Unit Type for a certain field. This helps to identify the unit that a certain value represents, like Seconds or Percentage

Enumerator

None	
Ticks	ticks
Seconds	seconds - secs
MilliSecs	millisecs - ms
Kilobytes	kilobytes - kB
Megabytes	megabytes - MB
Normalized	normalized - norm
Multiplier	multiplier - mult
Percentage	%
NormalizedPercentage	normalized % - n%
Degrees	degrees - \u00B0
PerSecond	per sec - /sec
DegreesPerSecond	\u00B0 / sec - \u00B0/sec
Radians	radians - rad
RadiansPerSecond	radian / sec - rad/s
TicksPerSecond	ticks / sec - tck/s
Units	units - units
Bytes	bytes - bytes
Count	count - count
Packets	packets - packets
Frames	frames - frames
FramesPerSecond	fps - fps
SquareMagnitude	sqrMagnitude - sqrMag

5.1.1.38 UnityPlayerLoopSystemAddMode

```
enum UnityPlayerLoopSystemAddMode [strong]
```

Enum representing the possible modes for adding a system to the Unity player loop.

Enumerator

FirstChild	Add the system as the first child of the parent system.
LastChild	Add the system as the last child of the parent system.
Before	Add the system before the parent system in the player loop.
After	Add the system after the parent system in the player loop.

5.1.2 Function Documentation

5.1.2.1 FusionGlobalScriptableObjectUnloadDelegate()

```
delegate void Fusion.FusionGlobalScriptableObjectUnloadDelegate (
    FusionGlobalScriptableObject instance )
```

A delegate that can be used to unload a [FusionGlobalScriptableObject](#).

5.1.2.2 NetworkObjectSpawnDelegate()

```
delegate void Fusion.NetworkObjectSpawnDelegate (
    NetworkSpawnOp result )
```

Network Object Spawn Delegate

5.1.2.3 RpcInvokeDelegate()

```
unsafe delegate void Fusion.RpcInvokeDelegate (
    NetworkBehaviour behaviour,
    SimulationMessage * message )
```

Represents a delegate that can be invoked by an RPC message.

Parameters

<i>behaviour</i>	The NetworkBehaviour associated with the RPC message.
<i>message</i>	The SimulationMessage associated with the RPC message.

The `RpcInvokeDelegate` is used to invoke an RPC message. The delegate is invoked by the `RpcSystem` when an RPC message is received. The delegate is invoked with the [NetworkBehaviour](#) associated with the RPC message and the [SimulationMessage](#) associated with the RPC message.

5.1.2.4 `RpcStaticInvokeDelegate()`

```
unsafe delegate void Fusion.RpcStaticInvokeDelegate (
    NetworkRunner runner,
    SimulationMessage * message )
```

Represents a delegate that can be invoked by an RPC message.

Parameters

<i>runner</i>	The NetworkRunner associated with the RPC message.
<i>message</i>	The SimulationMessage associated with the RPC message.

The `RpcInvokeDelegate` is used to invoke an RPC message. The delegate is invoked by the `RpcSystem` when an RPC message is received. The delegate is invoked with the [NetworkRunner](#) associated with the RPC message and the [SimulationMessage](#) associated with the RPC message.

5.2 Fusion.Analyzer Namespace Reference

Classes

- class [StaticConstructorAttribute](#)
Attribute to mark a constructor as a static constructor.
- class [StaticFieldAttribute](#)
Attribute to mark a static field with a reset mode.
- class [StaticFieldResetMethodAttribute](#)
Attribute to mark a method as a static field reset method.

Enumerations

- enum class [StaticFieldResetMode](#)
Specifies the reset mode for a static field.

5.2.1 Enumeration Type Documentation

5.2.1.1 `StaticFieldResetMode`

```
enum StaticFieldResetMode [strong]
```

Specifies the reset mode for a static field.

Enumerator

None	No reset mode.
Manual	Manual reset mode.
ResetMethod	Reset method mode.

5.3 Fusion.Async Namespace Reference

Classes

- class [TaskManager](#)

Task Factory is used to create new Tasks and Schedule long running Tasks

5.4 Fusion.Encryption Namespace Reference

Classes

- class [DataEncryptor](#)

Responsible for encrypting and decrypting data buffers

- interface [IDataEncryption](#)

Interface for classes that manage the encryption/decryption of byte arrays

5.5 Fusion.Internal Namespace Reference

Classes

- interface [IUnitySurrogate](#)

Represents an interface for Unity surrogates. This interface provides methods for reading and writing data.

- interface [IUnityValueSurrogate](#)

Represents an interface for Unity value surrogates. This interface provides a property for accessing and modifying the data.

- class [UnityArraySurrogate](#)

A base class for Unity array surrogates.

- class [UnityDictionarySurrogate](#)

A surrogate for serializing a dictionary.

- class [UnityLinkedListSurrogate](#)

A surrogate for serializing a linked list.

- class [UnitySurrogateBase](#)

Represents a base class for Unity surrogates. This class is serializable and provides abstract methods for reading, writing, and initializing data.

- class [UnityValueSurrogate](#)

Represents a base class for Unity value surrogates. This class is serializable and provides methods for reading, writing, and initializing data.

5.6 Fusion.LagCompensation Namespace Reference

Classes

- struct [AABB](#)
Represents an Axis-Aligned Bounding Box (AABB).
- class [BoxOverlapQuery](#)
Class that represents a box overlap query. Used to query against the [NetworkRunner.LagCompensation](#) API.
- struct [BoxOverlapQueryParams](#)
Base parameters needed to execute a box overlap query
- class [BVHDraw](#)
Provide a way to iterate over BVH and return a [BVHNodeDrawInfo](#) for each node.
- class [BVHNodeDrawInfo](#)
Container class to provide the necessary info to draw nodes from the BVH
- class [ColliderDrawInfo](#)
Container class to provide the necessary information to draw a hitbox collider
- class [HitboxColliderContainerDraw](#)
Provide a way to iterate over the [HitboxBuffer.HitboxSnapshot](#) and return the [ColliderDrawInfo](#) for each collider on the snapshot.
- class [LagCompensatedExt](#)
LagCompensated Extension methods
- class [LagCompensationDraw](#)
Provide access to iterate over the lag compensation system components and give the necessary information to draw them.
- struct [PositionRotationQueryParams](#)
Query parameters for position rotation query
- class [Query](#)
Base class for all Lag Compensation queries
- struct [QueryParams](#)
Base parameters needed to execute a query.
- class [RaycastAllQuery](#)
Class that represents a raycast all query. Used to query against the [NetworkRunner.LagCompensation](#) API.
- class [RaycastQuery](#)
Class that represents a raycast query. Used to query against the [NetworkRunner.LagCompensation](#) API.
- struct [RaycastQueryParams](#)
Base parameters needed to execute a raycast query
- class [SnapshotHistoryDraw](#)
Provide a way to iterate over the [HitboxBuffer](#) and return the [HitboxColliderContainerDraw](#) container for each snapshot on the buffer.
- class [SphereOverlapQuery](#)
Class that represents a sphere overlap query. Used to query against the [NetworkRunner.LagCompensation](#) API.
- struct [SphereOverlapQueryParams](#)
Base parameters needed to execute a sphere overlap query

Enumerations

- enum class [HitType](#)
Queries can hit either fusion's custom Hitbox or Unity's standard Physx/Box2D colliders.

Functions

- delegate void [PreProcessingDelegate](#) ([Query](#) query, [HashSet](#)< [HitboxRoot](#) > rootCandidates, [HashSet](#)< int > processedColliderIndices)

Pre-processing delegate for queries.

5.6.1 Enumeration Type Documentation

5.6.1.1 HitType

```
enum HitType [strong]
```

Queries can hit either fusion's custom [Hitbox](#) or Unity's standard Physx/Box2D colliders.

Enumerator

None	Used when a raycast does not hit anything. Not used on overlaps.
Hitbox	LagCompensatedHit is a Fusion Hitbox .
PhysX	LagCompensatedHit is a Unity PhysX Collider.
Box2D	LagCompensatedHit is a Unity Box2D Collider.

5.6.2 Function Documentation

5.6.2.1 PreProcessingDelegate()

```
delegate void Fusion.LagCompensation.PreProcessingDelegate (
    Query query,
    HashSet< HitboxRoot > rootCandidates,
    HashSet< int > processedColliderIndices )
```

Pre-processing delegate for queries.

Parameters

<i>query</i>	The query to be performed.
<i>rootCandidates</i>	The root candidates to be used for the query.
<i>processedColliderIndices</i>	The indices of the colliders that have been processed.

5.7 Fusion.Protocol Namespace Reference

Classes

- class [BitStream](#)
BitStream serialization methods.
- interface [ICommunicator](#)
Interface for a Communicator
- interface [IMessage](#)
Represents a Protocol Message

Enumerations

- enum class [DisconnectReason](#) : byte
List all Disconnect reason used by the Plugin to remove an Actor from the Room

5.7.1 Enumeration Type Documentation

5.7.1.1 DisconnectReason

```
enum DisconnectReason : byte [strong]
```

List all Disconnect reason used by the Plugin to remove an Actor from the Room

Enumerator

None	No reason
ServerLogic	Abstract disconnect reason
InvalidEventCode	Used when an event with other code other then the treated ones is received by the plugin
InvalidJoinMsgType	When the Join Message is not of the Request Type
InvalidJoinGameMode	When the Join Message does not contain a valid Game Mode
IncompatibleConfiguration	When any of the major settings of a message does not align with the current settings, like GameMode, Protocol Version, Serialization Version and Peer Mode
ServerAlreadyInRoom	When there is already a Server running on the current Room
Error	An error occured on the Plugin

5.8 Fusion.Runtime Namespace Reference

5.9 Fusion.Runtime.Unity Namespace Reference

5.10 Fusion.Sockets Namespace Reference

Classes

- interface [INetBitWriteStream](#)
Interface for writing bits to a stream.
- interface [INetPeerGroupCallbacks](#)
Defines the callbacks for network peer group events.
- interface [INetSocket](#)
Defines the interface for network socket operations.
- struct [NetAddress](#)
Represents a Network Address, which includes a IP and Port This can contains either a IPv4 or a IPv6 address
- struct [NetBitBuffer](#)
Represents a buffer for reading and writing bits.
- struct [NetBitBufferList](#)
Represents a linked list of [Fusion.Sockets.NetBitBuffer](#)
- struct [NetBitBufferNull](#)
Represents a null bit buffer for writing data.
- struct [NetBitBufferSerializer](#)
Represents a serializer for reading and writing data to a [NetBitBuffer](#).
- struct [NetCommandAccepted](#)
Accepted Command, sent by the server when a remote client connection is accepted
- struct [NetCommandConnect](#)
Connect Command used to signal a remote server that a client is trying to connect to it
- struct [NetCommandDisconnect](#)
Disconnect Command, it can be used by either side of the connection
- struct [NetCommandHeader](#)
Network Command Header Describe its type and usual settings for all commands
- struct [NetCommandRefused](#)
Refuse Command, sent by the server when the connection was refused. This happens when the server has reached its max connection capacity.
- struct [NetConfig](#)
General configuration used to drive the behavior of the Socket library
- struct [NetConfigNotify](#)
Represents the configuration for network notifications.
- struct [NetConfigSimulation](#)
Represents the configuration for network simulation.
- struct [NetConfigSimulationOscillator](#)
Represents an oscillator configuration for network simulation.
- struct [NetConnection](#)
Network connection
- struct [NetConnectionId](#)
Represents a network connection ID.
- struct [NetConnectionMap](#)
Represents a network connection map.

- struct [NetPeer](#)
Network Peer
- struct [NetPeerGroup](#)
Network Peer Group.
- struct [NetSendEnvelope](#)
Represents an envelope for sending network packets in the [Fusion.Sockets](#) namespace.
- struct [NetSocket](#)
Represents a network socket with a handle and a native socket.
- struct [ReliableHeader](#)
Represents a reliable header structure used in the [Fusion.Sockets](#) namespace.
- struct [ReliableId](#)
Represents a reliable identifier used in the [Fusion.Sockets](#) namespace.
- struct [ReliableKey](#)
Represents a reliable key structure used in the [Fusion.Sockets](#) namespace.
- struct [ReliableList](#)
Represents a list of reliable headers.

Enumerations

- enum class [NetCommands](#) : byte
Describe the Type of a Command Packet
- enum class [NetConnectFailedReason](#) : byte
The reason a connection with a remote server has failed
- enum class [NetConnectionStatus](#)
Represents the status of a network connection.
- enum class [NetDisconnectReason](#) : byte
Disconnect Reason Flag
- enum class [NetPacketType](#) : byte
Describe the type of a Networked Packet
- enum class [OnConnectionRequestReply](#)
Represents the possible replies to a connection request.

5.10.1 Enumeration Type Documentation

5.10.1.1 NetCommands

```
enum NetCommands : byte [strong]
```

Describe the Type of a Command Packet

5.10.1.2 NetConnectFailedReason

```
enum NetConnectFailedReason : byte [strong]
```

The reason a connection with a remote server has failed

Enumerator

Timeout	Server is not responding.
ServerFull	Server has accepted the max allowed Players
ServerRefused	Server refused the connection

5.10.1.3 NetConnectionStatus

```
enum NetConnectionStatus [strong]
```

Represents the status of a network connection.

Enumerator

Created	The connection has been created.
Connecting	The connection is in the process of connecting.
Connected	The connection is established.
Disconnected	The connection has been disconnected.
Shutdown	The connection has been shut down.

5.10.1.4 NetDisconnectReason

```
enum NetDisconnectReason : byte [strong]
```

Disconnect Reason Flag

Disconnect Reason Flag

Enumerator

Unknown	The reason for disconnection is unknown.
Timeout	The connection timed out.
Requested	The disconnection was requested.
SequenceOutOfBounds	The sequence is out of bounds.
SendWindowFull	The send window is full.
ByRemote	The disconnection was initiated by the remote party.

5.10.1.5 NetPacketType

```
enum NetPacketType : byte [strong]
```

Describe the type of a Networked Packet

5.10.1.6 OnConnectionRequestReply

```
enum OnConnectionRequestReply [strong]
```

Represents the possible replies to a connection request.

Enumerator

Ok	The connection request is accepted.
Refuse	The connection request is refused.
Waiting	The connection request is waiting for a decision.

5.11 Fusion.Sockets.Stun Namespace Reference

Enumerations

- enum class [NATType](#) : byte
Specifies UDP network type.

5.11.1 Enumeration Type Documentation

5.11.1.1 NATType

```
enum NATType : byte [strong]
```

Specifies UDP network type.

Enumerator

Invalid	Invalid NAT Type
UdpBlocked	UDP is always blocked.
OpenInternet	No NAT, public IP, no firewall.
FullCone	A full cone NAT is one where all requests from the same internal IP address and port are mapped to the same external IP address and port. Furthermore, any external host can send a packet to the internal host, by sending a packet to the mapped external address.
Symmetric	A symmetric NAT is one where all requests from the same internal IP address and port, to a specific destination IP address and port, are mapped to the same external IP address and port. If the same host sends a packet with the same source address and port, but to a different destination, a different mapping is used. Furthermore, only the external host that receives a packet can send a UDP packet back to the internal host.

5.12 Fusion.Statistics Namespace Reference

Classes

- class [BehaviourStatisticsManager](#)
Behaviour statistics manager will provide access to the behaviour statistics snapshots.
- class [BehaviourStatisticsSnapshot](#)
Represents a snapshot of statistics related to Fusion Behaviour type execution.
- class [FusionStatisticsManager](#)
Represents a fusion statistics manager.
- class [FusionStatisticsSnapshot](#)
Represents a snapshot of Fusion statistics.
- class [LagCompensationStatisticsSnapshot](#)
Represents a snapshot of lag compensation statistics.
- struct [MemoryStatisticsSnapshot](#)
Represents a snapshot of specific memory statistics. For basic total allocated memory and total free memory check the Fusion Statistics.
- class [NetworkObjectStatisticsManager](#)
Manages network object statistics for monitored network objects.
- class [NetworkObjectStatisticsSnapshot](#)
Represents a snapshot of network object statistics.

5.13 UnityEngine Namespace Reference

5.14 UnityEngine.SceneManagement Namespace Reference

5.15 UnityEngine.Scripting Namespace Reference

5.16 UnityEngine.Serialization Namespace Reference

Chapter 6

Class Documentation

6.1 `_128` Struct Reference

A `FixedStorage` that can hold up to 128 words.

Inherits `INetworkStruct`, and `IFixedStorage`.

Public Attributes

- fixed uint `Data` [128]
The data of the `FixedStorage`.

Static Public Attributes

- const int `SIZE` = 512
The size of the `FixedStorage` in bytes.

6.1.1 Detailed Description

A `FixedStorage` that can hold up to 128 words.

6.1.2 Member Data Documentation

6.1.2.1 Data

```
fixed uint Data[128]
```

The data of the `FixedStorage`.

6.1.2.2 SIZE

```
const int SIZE = 512 [static]
```

The size of the [FixedStorage](#) in bytes.

6.2 [_16 Struct Reference](#)

A [FixedStorage](#) that can hold up to 16 words.

Inherits [INetworkStruct](#), and [IFixedStorage](#).

Public Attributes

- fixed uint [Data](#) [16]
The data of the [FixedStorage](#).

Static Public Attributes

- const int [SIZE](#) = 64
The size of the [FixedStorage](#) in bytes.

6.2.1 Detailed Description

A [FixedStorage](#) that can hold up to 16 words.

6.2.2 Member Data Documentation

6.2.2.1 Data

```
fixed uint Data[16]
```

The data of the [FixedStorage](#).

6.2.2.2 SIZE

```
const int SIZE = 64 [static]
```

The size of the [FixedStorage](#) in bytes.

6.3 [_2 Struct Reference](#)

A [FixedStorage](#) that can hold up to 2 words.

Inherits [INetworkStruct](#), and [IFixedStorage](#).

Public Attributes

- fixed uint [Data](#) [2]
The data of the [FixedStorage](#).

Static Public Attributes

- const int [SIZE](#) = 8
The size of the [FixedStorage](#) in bytes.

6.3.1 Detailed Description

A [FixedStorage](#) that can hold up to 2 words.

6.3.2 Member Data Documentation

6.3.2.1 Data

```
fixed uint Data[2]
```

The data of the [FixedStorage](#).

6.3.2.2 SIZE

```
const int SIZE = 8 [static]
```

The size of the [FixedStorage](#) in bytes.

6.4 [_256 Struct Reference](#)

A [FixedStorage](#) that can hold up to 256 words.

Inherits [INetworkStruct](#), and [IFixedStorage](#).

Public Attributes

- fixed uint [Data](#) [256]
The data of the [FixedStorage](#).

Static Public Attributes

- const int [SIZE](#) = 1024
The size of the [FixedStorage](#) in bytes.

6.4.1 Detailed Description

A [FixedStorage](#) that can hold up to 256 words.

6.4.2 Member Data Documentation

6.4.2.1 Data

```
fixed uint Data[256]
```

The data of the [FixedStorage](#).

6.4.2.2 SIZE

```
const int SIZE = 1024 [static]
```

The size of the [FixedStorage](#) in bytes.

6.5 [_32](#) Struct Reference

A [FixedStorage](#) that can hold up to 32 words.

Inherits [INetworkStruct](#), and [IFixedStorage](#).

Public Attributes

- fixed uint [Data](#) [32]
The data of the [FixedStorage](#).

Static Public Attributes

- const int [SIZE](#) = 128
The size of the [FixedStorage](#) in bytes.

6.5.1 Detailed Description

A [FixedStorage](#) that can hold up to 32 words.

6.5.2 Member Data Documentation

6.5.2.1 Data

```
fixed uint Data[32]
```

The data of the [FixedStorage](#).

6.5.2.2 SIZE

```
const int SIZE = 128 [static]
```

The size of the [FixedStorage](#) in bytes.

6.6 _4 Struct Reference

A [FixedStorage](#) that can hold up to 4 words.

Inherits [INetworkStruct](#), and [IFixedStorage](#).

Public Attributes

- fixed uint [Data](#) [4]
The data of the [FixedStorage](#).

Static Public Attributes

- const int [SIZE](#) = 16
The size of the [FixedStorage](#) in bytes.

6.6.1 Detailed Description

A [FixedStorage](#) that can hold up to 4 words.

6.6.2 Member Data Documentation

6.6.2.1 Data

```
fixed uint Data[4]
```

The data of the [FixedStorage](#).

6.6.2.2 SIZE

```
const int SIZE = 16 [static]
```

The size of the [FixedStorage](#) in bytes.

6.7 [_512 Struct Reference](#)

A [FixedStorage](#) that can hold up to 512 words.

Inherits [INetworkStruct](#), and [IFixedStorage](#).

Public Attributes

- fixed uint [Data](#) [512]
The data of the [FixedStorage](#).

Static Public Attributes

- const int [SIZE](#) = 2048
The size of the [FixedStorage](#) in bytes.

6.7.1 Detailed Description

A [FixedStorage](#) that can hold up to 512 words.

6.7.2 Member Data Documentation

6.7.2.1 Data

```
fixed uint Data[512]
```

The data of the [FixedStorage](#).

6.7.2.2 SIZE

```
const int SIZE = 2048 [static]
```

The size of the [FixedStorage](#) in bytes.

6.8 _64 Struct Reference

A [FixedStorage](#) that can hold up to 64 words.

Inherits [INetworkStruct](#), and [IFixedStorage](#).

Public Attributes

- fixed uint [Data](#) [64]
The data of the [FixedStorage](#).

Static Public Attributes

- const int [SIZE](#) = 256
The size of the [FixedStorage](#) in bytes.

6.8.1 Detailed Description

A [FixedStorage](#) that can hold up to 64 words.

6.8.2 Member Data Documentation

6.8.2.1 Data

```
fixed uint Data[64]
```

The data of the [FixedStorage](#).

6.8.2.2 SIZE

```
const int SIZE = 256 [static]
```

The size of the [FixedStorage](#) in bytes.

6.9 [_8 Struct Reference](#)

A [FixedStorage](#) that can hold up to 8 words.

Inherits [INetworkStruct](#), and [IFixedStorage](#).

Public Attributes

- fixed uint [Data](#) [8]
The data of the [FixedStorage](#).

Static Public Attributes

- const int [SIZE](#) = 32
The size of the [FixedStorage](#) in bytes.

6.9.1 Detailed Description

A [FixedStorage](#) that can hold up to 8 words.

6.9.2 Member Data Documentation

6.9.2.1 Data

```
fixed uint Data[8]
```

The data of the [FixedStorage](#).

6.9.2.2 SIZE

```
const int SIZE = 32 [static]
```

The size of the [FixedStorage](#) in bytes.

6.10 Allocator Struct Reference

Memory [Allocator](#)

Classes

- struct [Config](#)
Memory [Allocator](#) Configuration

Public Member Functions

- Block * **GetBlock** (int index)
- Block * **GetBlock** (long index)
- int **GetBlockBucket** (long index)
- Block * **GetBlockForPointer** (void *ptr)
- int **GetBlockIndexForPointer** (void *ptr)
- byte * **GetBlockMemory** (Block *block)
- byte * **GetBlockMemory** (long blockIndex)
- Bucket * **GetBucket** (int index)
- Bucket * **GetBucketForBlock** (Block *block)
- BlockList * **GetBucketList** (int index)
- bool **IsPointerInMeta** (void *p)
- void * **Meta** ([Ptr](#) ptr)
- [Ptr](#) **Meta** (void *p)

Static Public Member Functions

- static void **DebugVerifyBucketIntegrity** ([Allocator](#) *a, int index)
- static void * **TryAllocateSegmentFromBlock** ([Allocator](#) *a, Bucket *bucket, Block *block, int size)
- static int **WordCount** (int size)

Public Attributes

- Block * **_blocks**
- BlockList * **_blocksFreeList**
- Bucket * **_buckets**
- BlockList * **_bucketsLists**
- byte * **_bucketsMap**
- void * **_checksum**
- int **_checksumByteLength**
- [Config](#) **_config**
- void * **_globals**
- byte * **_heap**
- int **_maxBlockIndexUsed**
- byte * **_meta**
- void * **_replicate**
- int **_replicateByteLength**
- byte * **_root**

Static Public Attributes

- const int `BUCKET_COUNT` = 57
Bucket Count
- const byte `BUCKET_INVALID` = 255
Bucket Invalid
- const int `HEAP_ALIGNMENT` = 8
Heap Alignment
- const int `PTR_SIZE` = 8
- const int `REPLICATE_WORD_ALIGN` = `REPLICATE_WORD_SIZE`
Replicate Word Align
- const int `REPLICATE_WORD_SHIFT` = 2
Replicate Word Shift
- const int `REPLICATE_WORD_SIZE` = 1 << `REPLICATE_WORD_SHIFT`
Replicate Word Size
- const int `SIZE` = 112
- const int `WORD_BYTE_SIZE` = 1 << 3
- const int `WORD_SHIFT` = 3

6.10.1 Detailed Description

Memory [Allocator](#)

6.10.2 Member Data Documentation

6.10.2.1 BUCKET_COUNT

```
const int BUCKET_COUNT = 57 [static]
```

Bucket Count

6.10.2.2 BUCKET_INVALID

```
const byte BUCKET_INVALID = 255 [static]
```

Bucket Invalid

6.10.2.3 HEAP_ALIGNMENT

```
const int HEAP_ALIGNMENT = 8 [static]
```

Heap Alignment

6.10.2.4 REPLICATE_WORD_ALIGN

```
const int REPLICATE_WORD_ALIGN = REPLICATE_WORD_SIZE [static]
```

Replicate Word Align

6.10.2.5 REPLICATE_WORD_SHIFT

```
const int REPLICATE_WORD_SHIFT = 2 [static]
```

Replicate Word Shift

6.10.2.6 REPLICATE_WORD_SIZE

```
const int REPLICATE_WORD_SIZE = 1 << REPLICATE_WORD_SHIFT [static]
```

Replicate Word Size

6.11 Allocator.Config Struct Reference

Memory [Allocator](#) Configuration

Public Member Functions

- [Config](#) ([PageSizes](#) shift, int count, int globalsSize)
Config Constructor
- bool [Equals](#) ([Config](#) other)
Check Config equality
- override bool [Equals](#) (object obj)
Check Config equality
- override int [GetHashCode](#) ()
Get Hash Code
- override string [ToString](#) ()
Config ToString

Public Attributes

- int [BlockCount](#)
Block Count Config value
- int [BlockShift](#)
Block Shift Config value
- int [GlobalsSize](#)
Globals Size Config value

Static Public Attributes

- const int [DEFAULT_BLOCK_COUNT](#) = 256
Default Block Count
- const [PageSizes](#) [DEFAULT_BLOCK_SHIFT](#) = [PageSizes._32Kb](#)
Default Block Shift
- const int [SIZE](#) = 12
Config Size

Properties

- int [BlockByteSize](#) [get]
Block Size in Bytes
- int [BlockWordCount](#) [get]
Block Size in Words
- int [HeapSizeAllocated](#) [get]
Heap Size Allocated in Bytes
- int [HeapSizeUsable](#) [get]
Heap Size in Bytes

6.11.1 Detailed Description

Memory [Allocator](#) Configuration

6.11.2 Constructor & Destructor Documentation

6.11.2.1 Config()

```
Config (
    PageSizes shift,
    int count,
    int globalsSize )
```

[Config](#) Constructor

Parameters

<i>shift</i>	Block Shift
<i>count</i>	Block Count
<i>globalsSize</i>	Globals Size

6.11.3 Member Function Documentation

6.11.3.1 Equals() [1/2]

```
bool Equals (
    Config other )
```

Check [Config](#) equality

Parameters

<i>other</i>	Config Ref
--------------	----------------------------

Returns

True if has the same values

6.11.3.2 Equals() [2/2]

```
override bool Equals (
    object obj )
```

Check [Config](#) equality

Parameters

<i>obj</i>	Any object reference
------------	----------------------

Returns

True if *obj* is a [Config](#) and has the same values

6.11.3.3 GetHashCode()

```
override int GetHashCode ( )
```

Get Hash Code

Returns

Hash Code

6.11.3.4 ToString()

```
override string ToString ( )
```

[Config](#) ToString

6.11.4 Member Data Documentation

6.11.4.1 BlockCount

```
int BlockCount
```

Block Count [Config](#) value

6.11.4.2 BlockShift

```
int BlockShift
```

Block Shift [Config](#) value

6.11.4.3 DEFAULT_BLOCK_COUNT

```
const int DEFAULT_BLOCK_COUNT = 256 [static]
```

Default Block Count

6.11.4.4 DEFAULT_BLOCK_SHIFT

```
const PageSizes DEFAULT_BLOCK_SHIFT = PageSizes._32Kb [static]
```

Default Block Shift

6.11.4.5 GlobalsSize

```
int GlobalsSize
```

Globals Size [Config](#) value

6.11.4.6 SIZE

```
const int SIZE = 12 [static]
```

[Config Size](#)

6.11.5 Property Documentation

6.11.5.1 BlockByteSize

```
int BlockByteSize [get]
```

Block Size in Bytes

6.11.5.2 BlockWordCount

```
int BlockWordCount [get]
```

Block Size in Words

6.11.5.3 HeapSizeAllocated

```
int HeapSizeAllocated [get]
```

Heap Size Allocated in Bytes

6.11.5.4 HeapSizeUsable

```
int HeapSizeUsable [get]
```

Heap Size in Bytes

6.12 StaticConstructorAttribute Class Reference

Attribute to mark a constructor as a static constructor.

Inherits Attribute.

6.12.1 Detailed Description

Attribute to mark a constructor as a static constructor.

6.13 StaticFieldAttribute Class Reference

Attribute to mark a static field with a reset mode.

Inherits Attribute.

Public Member Functions

- [StaticFieldAttribute](#) ()
Initializes a new instance of the [StaticFieldAttribute](#) class with the default reset mode.
- [StaticFieldAttribute](#) ([StaticFieldResetMode](#) resetMode)
Initializes a new instance of the [StaticFieldAttribute](#) class with the specified reset mode.

Properties

- [StaticFieldResetMode](#) [Reset](#) [get]
Gets the reset mode for the static field.

6.13.1 Detailed Description

Attribute to mark a static field with a reset mode.

6.13.2 Constructor & Destructor Documentation

6.13.2.1 StaticFieldAttribute() [1/2]

```
StaticFieldAttribute (  
    StaticFieldResetMode resetMode )
```

Initializes a new instance of the [StaticFieldAttribute](#) class with the specified reset mode.

Parameters

<code>resetMode</code>	The reset mode for the static field.
------------------------	--------------------------------------

6.13.2.2 StaticFieldAttribute() [2/2]

```
StaticFieldAttribute ( )
```

Initializes a new instance of the [StaticFieldAttribute](#) class with the default reset mode.

6.13.3 Property Documentation

6.13.3.1 Reset

```
StaticFieldResetMode Reset [get]
```

Gets the reset mode for the static field.

6.14 StaticFieldResetMethodAttribute Class Reference

Attribute to mark a method as a static field reset method.

Inherits Attribute.

Public Member Functions

- [StaticFieldResetMethodAttribute](#) ()
Initializes a new instance of the [StaticFieldResetMethodAttribute](#) class.
- [StaticFieldResetMethodAttribute](#) (bool calledAutomatically)
Initializes a new instance of the [StaticFieldResetMethodAttribute](#) class with the specified automatic call flag.

6.14.1 Detailed Description

Attribute to mark a method as a static field reset method.

6.14.2 Constructor & Destructor Documentation

6.14.2.1 StaticFieldResetMethodAttribute() [1/2]

```
StaticFieldResetMethodAttribute (  
    bool calledAutomatically )
```

Initializes a new instance of the [StaticFieldResetMethodAttribute](#) class with the specified automatic call flag.

Parameters

<code>calledAutomatically</code>	Indicates whether the method is called automatically.
----------------------------------	---

6.14.2.2 `StaticFieldResetMethodAttribute()` [2/2]

`StaticFieldResetMethodAttribute` ()

Initializes a new instance of the `StaticFieldResetMethodAttribute` class.

6.15 Angle Struct Reference

A Networked fusion type for degrees. This can be used with the `NetworkedAttribute`, in RPCs, or in `NetworkInput` structs.

Inherits `INetworkStruct`, and `IEquatable< Angle >`.

Public Member Functions

- void `Clamp` (`Angle` min, `Angle` max)
Clamps the current value to the supplied min-max range.
- bool `Equals` (`Angle` other)
Checks equality with another `Angle`.
- override bool `Equals` (object obj)
Checks equality with an object.
- override int `GetHashCode` ()
Gets the hash code.
- override string `ToString` ()
String representation of the `Angle`.

Static Public Member Functions

- static `Angle Clamp` (`Angle` value, `Angle` min, `Angle` max)
Returns a the value, clamped to the min-max range.
- static `Angle Lerp` (`Angle` a, `Angle` b, float t)
Lerps between two angle values.
- static `Angle Max` (`Angle` a, `Angle` b)
Returns the larger of two supplied angles.
- static `Angle Min` (`Angle` a, `Angle` b)
Returns the smaller of two supplied angles.
- static implicit operator `Angle` (double value)
Converts double to `Angle`.
- static implicit operator `Angle` (float value)
Converts float to `Angle`.
- static implicit operator `Angle` (int value)

- Converts int to [Angle](#).*

 - static [operator double](#) ([Angle](#) value)
- Converts [Angle](#) to double.*

 - static [operator float](#) ([Angle](#) value)
- Converts [Angle](#) to float.*

 - static bool [operator!=](#) ([Angle](#) a, [Angle](#) b)
- Inequality operator for [Angle](#) struct.*

 - static [Angle operator+](#) ([Angle](#) a, [Angle](#) b)
- Addition operator for [Angle](#) struct.*

 - static [Angle operator-](#) ([Angle](#) a, [Angle](#) b)
- Subtraction operator for [Angle](#) struct.*

 - static bool [operator<](#) ([Angle](#) a, [Angle](#) b)
- Less than operator for [Angle](#) struct.*

 - static bool [operator<=](#) ([Angle](#) a, [Angle](#) b)
- Less than or equal to operator for [Angle](#) struct.*

 - static bool [operator==](#) ([Angle](#) a, [Angle](#) b)
- Equality operator for [Angle](#) struct.*

 - static bool [operator>](#) ([Angle](#) a, [Angle](#) b)
- Greater than operator for [Angle](#) struct.*

 - static bool [operator>=](#) ([Angle](#) a, [Angle](#) b)
- Greater than or equal to operator for [Angle](#) struct.*

Public Attributes

- int [_value](#)

Static Public Attributes

- const int [_360](#) = 360 * [ACCURACY](#)
- const int [ACCURACY](#) = 10000
- const int [DECIMALS](#) = 4
- const int [SIZE](#) = 4

Size of this struct in bytes.

6.15.1 Detailed Description

A Networked fusion type for degrees. This can be used with the [NetworkedAttribute](#), in RPCs, or in [NetworkInput](#) structs.

6.15.2 Member Function Documentation

6.15.2.1 Clamp() [1/2]

```
void Clamp (
    Angle min,
    Angle max )
```

Clamps the current value to the supplied min-max range.

6.15.2.2 Clamp() [2/2]

```
static Angle Clamp (
    Angle value,
    Angle min,
    Angle max ) [static]
```

Returns a the value, clamped to the min-max range.

6.15.2.3 Equals() [1/2]

```
bool Equals (
    Angle other )
```

Checks equality with another [Angle](#).

Parameters

<i>other</i>	Other Angle .
--------------	-------------------------------

Returns

Equality result.

6.15.2.4 Equals() [2/2]

```
override bool Equals (
    object obj )
```

Checks equality with an object.

Parameters

<i>obj</i>	Object to compare.
------------	--------------------

Returns

Equality result.

6.15.2.5 GetHashCode()

```
override int GetHashCode ( )
```

Gets the hash code.

Returns

Hash code.

6.15.2.6 Lerp()

```
static Angle Lerp (
    Angle a,
    Angle b,
    float t ) [static]
```

Lerps between two angle values.

6.15.2.7 Max()

```
static Angle Max (
    Angle a,
    Angle b ) [static]
```

Returns the larger of two supplied angles.

6.15.2.8 Min()

```
static Angle Min (
    Angle a,
    Angle b ) [static]
```

Returns the smaller of two supplied angles.

6.15.2.9 operator Angle() [1/3]

```
static implicit operator Angle (
    double value ) [static]
```

Converts double to [Angle](#).

Parameters

<i>value</i>	Double value.
--------------	---------------

Returns

[Angle](#) instance with the value of the double.

6.15.2.10 operator Angle() [2/3]

```
static implicit operator Angle (  
    float value ) [static]
```

Converts float to [Angle](#).

Parameters

<i>value</i>	Float value.
--------------	--------------

Returns

[Angle](#) instance with the value of the float.

6.15.2.11 operator Angle() [3/3]

```
static implicit operator Angle (  
    int value ) [static]
```

Converts int to [Angle](#).

Parameters

<i>value</i>	Integer value.
--------------	----------------

Returns

[Angle](#) instance with the value of the integer.

6.15.2.12 operator double()

```
static operator double (  
    Angle value ) [explicit], [static]
```


Converts [Angle](#) to double.

Parameters

<i>value</i>	Angle instance.
--------------	---------------------------------

Returns

Double representation of the [Angle](#).

6.15.2.13 operator float()

```
static operator float (  
    Angle value ) [explicit], [static]
```

Converts [Angle](#) to float.

Parameters

<i>value</i>	Angle instance.
--------------	---------------------------------

Returns

Float representation of the [Angle](#).

6.15.2.14 operator"!="()

```
static bool operator!= (  
    Angle a,  
    Angle b ) [static]
```

Inequality operator for [Angle](#) struct.

Parameters

<i>a</i>	First Angle instance.
<i>b</i>	Second Angle instance.

Returns

True if the value of the first [Angle](#) instance is not equal to the value of the second [Angle](#) instance, otherwise false.

6.15.2.15 operator+()

```
static Angle operator+ (  
    Angle a,  
    Angle b ) [static]
```

Addition operator for [Angle](#) struct.

Parameters

<i>a</i>	First Angle instance.
<i>b</i>	Second Angle instance.

Returns

A new [Angle](#) instance that is the sum of the first and second [Angle](#) instances, wrapped around at 360 degrees if necessary.

6.15.2.16 operator-()

```
static Angle operator- (  
    Angle a,  
    Angle b ) [static]
```

Subtraction operator for [Angle](#) struct.

Parameters

<i>a</i>	First Angle instance.
<i>b</i>	Second Angle instance.

Returns

A new [Angle](#) instance that is the difference of the first and second [Angle](#) instances, wrapped around at 360 degrees if necessary.

6.15.2.17 operator<()

```
static bool operator< (  
    Angle a,  
    Angle b ) [static]
```

Less than operator for [Angle](#) struct.

Parameters

<i>a</i>	First Angle instance.
<i>b</i>	Second Angle instance.

Returns

True if the value of the first [Angle](#) instance is less than the value of the second [Angle](#) instance, otherwise false.

6.15.2.18 operator<=()

```
static bool operator<= (
    Angle a,
    Angle b ) [static]
```

Less than or equal to operator for [Angle](#) struct.

Parameters

<i>a</i>	First Angle instance.
<i>b</i>	Second Angle instance.

Returns

True if the value of the first [Angle](#) instance is less than or equal to the value of the second [Angle](#) instance, otherwise false.

6.15.2.19 operator==()

```
static bool operator==(
    Angle a,
    Angle b ) [static]
```

Equality operator for [Angle](#) struct.

Parameters

<i>a</i>	First Angle instance.
<i>b</i>	Second Angle instance.

Returns

True if the value of the first [Angle](#) instance is equal to the value of the second [Angle](#) instance, otherwise false.

6.15.2.20 operator>()

```
static bool operator> (
    Angle a,
    Angle b ) [static]
```

Greater than operator for [Angle](#) struct.

Parameters

<i>a</i>	First Angle instance.
<i>b</i>	Second Angle instance.

Returns

True if the value of the first [Angle](#) instance is greater than the value of the second [Angle](#) instance, otherwise false.

6.15.2.21 operator>=()

```
static bool operator>= (
    Angle a,
    Angle b ) [static]
```

Greater than or equal to operator for [Angle](#) struct.

Parameters

<i>a</i>	First Angle instance.
<i>b</i>	Second Angle instance.

Returns

True if the value of the first [Angle](#) instance is greater than or equal to the value of the second [Angle](#) instance, otherwise false.

6.15.2.22 ToString()

```
override string ToString ( )
```

String representation of the [Angle](#).

6.15.3 Member Data Documentation

6.15.3.1 SIZE

```
const int SIZE = 4 [static]
```

Size of this struct in bytes.

6.16 ArrayLengthAttribute Class Reference

Editor attribute for selecting the minimum and maximum length constraints for an array field.

Inherits [DecoratingPropertyAttribute](#).

Public Member Functions

- [ArrayLengthAttribute](#) (int length)
Initializes a new instance of the [ArrayLengthAttribute](#) class with the specified length.
- [ArrayLengthAttribute](#) (int minLength, int maxLength)
Initializes a new instance of the [ArrayLengthAttribute](#) class with the specified minimum and maximum lengths.

Properties

- int [MaxLength](#) [get]
Gets the maximum length of the array.
- int [MinLength](#) [get]
Gets the minimum length of the array.

Additional Inherited Members

6.16.1 Detailed Description

Editor attribute for selecting the minimum and maximum length constraints for an array field.

6.16.2 Constructor & Destructor Documentation

6.16.2.1 ArrayLengthAttribute() [1/2]

```
ArrayLengthAttribute (  
    int length )
```

Initializes a new instance of the [ArrayLengthAttribute](#) class with the specified length.

Parameters

<i>length</i>	The length of the array.
---------------	--------------------------

6.16.2.2 ArrayLengthAttribute() [2/2]

```
ArrayLengthAttribute (  
    int minLength,  
    int maxLength )
```

Initializes a new instance of the [ArrayLengthAttribute](#) class with the specified minimum and maximum lengths.

Parameters

<i>minLength</i>	The minimum length of the array.
<i>maxLength</i>	The maximum length of the array.

6.16.3 Property Documentation**6.16.3.1 MaxLength**

```
int MaxLength [get]
```

Gets the maximum length of the array.

6.16.3.2 MinLength

```
int MinLength [get]
```

Gets the minimum length of the array.

6.17 AssemblyNameAttribute Class Reference

Specifies that the attributed field represents the name of an assembly.

Inherits [DrawerPropertyAttribute](#).

Properties

- bool [RequiresUnsafeCode](#) [get, set]
Gets or sets a value indicating whether the assembly requires unsafe code.

6.17.1 Detailed Description

Specifies that the attributed field represents the name of an assembly.

6.17.2 Property Documentation

6.17.2.1 RequiresUnsafeCode

```
bool RequiresUnsafeCode [get], [set]
```

Gets or sets a value indicating whether the assembly requires unsafe code.

6.18 AssetObject Class Reference

Base class for all [Fusion](#) assets.

Inherits [ScriptableObject](#).

6.18.1 Detailed Description

Base class for all [Fusion](#) assets.

6.19 TaskManager Class Reference

Task Factory is used to create new Tasks and Schedule long running Tasks

Static Public Member Functions

- static Task [ContinueWhenAll](#) (Task[] precedingTasks, Func< CancellationToken, Task > action, CancellationToken cancellationToken)
Run a continuation Task after all other Tasks have completed
- static async Task [Delay](#) (int delay, CancellationToken token=default)
Custom Task Delay method as Task.Delay is not supported by WebGL Builds
- static Task [Run](#) (Func< CancellationToken, Task > action, CancellationToken cancellationToken, Task↔ CreationOptions options=TaskCreationOptions.None)
Run an Action asynchronously
- static Task [Service](#) (Action recurringAction, CancellationToken cancellationToken, int interval, string serviceName=null)
Starts a service task that will invoke a recurring action at specified intervals.
- static Task [Service](#) (Func< Task< bool >> recurringAction, CancellationToken cancellationToken, int interval, string serviceName=null)
Start a Service Task that will invoke a Recurring Action every each interval in millis
- static void [Setup](#) ()
Setup a new TaskFactory tailored to work with Unity

6.19.1 Detailed Description

Task Factory is used to create new Tasks and Schedule long running Tasks

6.19.2 Member Function Documentation

6.19.2.1 ContinueWhenAll()

```
static Task ContinueWhenAll (
    Task[] precedingTasks,
    Func< CancellationToken, Task > action,
    CancellationToken cancellationToken ) [static]
```

Run a continuation Task after all other Tasks have completed

Parameters

<i>precedingTasks</i>	List of pending tasks to wait
<i>action</i>	Action to run after the Tasks
<i>cancellationToken</i>	ellationToken used to stop the Action

Returns

[Async](#) Task based on the Action

6.19.2.2 Delay()

```
static async Task Delay (
    int delay,
    CancellationToken token = default ) [static]
```

Custom Task Delay method as Task.Delay is not supported by WebGL Builds

Parameters

<i>delay</i>	Delay in milliseconds to wait
<i>token</i>	Cancellation Token used to stop the Delay

Returns

Awaitable Task

6.19.2.3 Run()

```
static Task Run (
    Func< CancellationToken, Task > action,
    CancellationToken cancellationToken,
    TaskCreationOptions options = TaskCreationOptions.None ) [static]
```

Run an Action asynchronously

Parameters

<i>action</i>	Action to be invoked
<i>cancellationToken</i>	Cancellation token used to stop the Action
<i>options</i>	Extra Task Creation options

Returns

[Async](#) Task based on the Action

6.19.2.4 Service() [1/2]

```
static Task Service (
    Action recurringAction,
    CancellationToken cancellationToken,
    int interval,
    string serviceName = null ) [static]
```

Starts a service task that will invoke a recurring action at specified intervals.

Parameters

<i>recurringAction</i>	The action to be invoked at each interval.
<i>cancellationToken</i>	The cancellation token used to stop the service.
<i>interval</i>	The interval in milliseconds between each invocation of the action.
<i>serviceName</i>	An optional custom name for the service.

Returns

A task representing the service.

6.19.2.5 Service() [2/2]

```
static Task Service (
    Func< Task< bool >> recurringAction,
    CancellationToken cancellationToken,
```

```
int interval,  
string serviceName = null ) [static]
```

Start a Service Task that will invoke a Recurring Action every each interval in millis

Parameters

<i>recurringAction</i>	Action invoked every interval. It can return false to stop the service
<i>cancellationToken</i>	CancellationToken used to stop the service
<i>interval</i>	Interval between action invoke
<i>serviceName</i>	Custom id name for the Service

Returns

Service Task

6.19.2.6 Setup()

```
static void Setup ( ) [static]
```

Setup a new TaskFactory tailored to work with Unity

6.20 AtomicInt Struct Reference

Represents an atomic integer that provides thread-safe operations.

Public Member Functions

- [AtomicInt](#) (int value)
Initializes a new instance of the AtomicInt struct with the specified value.
- int [CompareExchange](#) (int value, int assumed)
Compares the current value with a specified value and, if they are equal, replaces the current value.
- int [Decrement](#) ()
Atomically decrements the current value by one and returns the decremented value.
- int [Exchange](#) (int value)
Atomically sets the value to the specified value and returns the original value.
- int [IncrementPost](#) ()
Atomically increments the current value by one and returns the original value.
- int [IncrementPre](#) ()
Atomically increments the current value by one and returns the incremented value.

Public Attributes

- volatile int [_value](#)
The underlying value of the atomic integer.

Properties

- int [Value](#) [get]
Gets the current value of the atomic integer.

6.20.1 Detailed Description

Represents an atomic integer that provides thread-safe operations.

6.20.2 Constructor & Destructor Documentation

6.20.2.1 AtomicInt()

```
AtomicInt (
    int value )
```

Initializes a new instance of the [AtomicInt](#) struct with the specified value.

Parameters

<i>value</i>	The initial value of the atomic integer.
--------------	--

6.20.3 Member Function Documentation

6.20.3.1 CompareExchange()

```
int CompareExchange (
    int value,
    int assumed )
```

Compares the current value with a specified value and, if they are equal, replaces the current value.

Parameters

<i>value</i>	The value to set if the comparison succeeds.
<i>assumed</i>	The value to compare to the current value.

Returns

The original value before the comparison.

6.20.3.2 Decrement()

```
int Decrement ( )
```

Atomically decrements the current value by one and returns the decremented value.

Returns

The decremented value.

6.20.3.3 Exchange()

```
int Exchange (  
    int value )
```

Atomically sets the value to the specified value and returns the original value.

Parameters

<i>value</i>	The value to set.
--------------	-------------------

Returns

The original value before the exchange.

6.20.3.4 IncrementPost()

```
int IncrementPost ( )
```

Atomically increments the current value by one and returns the original value.

Returns

The original value before the increment.

6.20.3.5 IncrementPre()

```
int IncrementPre ( )
```

Atomically increments the current value by one and returns the incremented value.

Returns

The incremented value.

6.20.4 Member Data Documentation

6.20.4.1 `_value`

```
volatile int _value
```

The underlying value of the atomic integer.

6.20.5 Property Documentation

6.20.5.1 Value

```
int Value [get]
```

Gets the current value of the atomic integer.

6.21 AuthorityMasks Class Reference

Provides constants and methods for managing authority masks.

Static Public Attributes

- const int `ALL` = `STATE` | `INPUT` | `PROXY`
Constant representing all authorities.
- const int `INPUT` = `1 << 1`
Constant representing the input authority mask.
- const int `NONE` = `0`
Constant representing no authority.
- const int `PROXY` = `1 << 2`
Constant representing the proxy authority mask.
- const int `STATE` = `1 << 0`
Constant representing the state authority mask.

6.21.1 Detailed Description

Provides constants and methods for managing authority masks.

6.21.2 Member Data Documentation

6.21.2.1 ALL

```
const int ALL = STATE | INPUT | PROXY [static]
```

Constant representing all authorities.

6.21.2.2 INPUT

```
const int INPUT = 1 << 1 [static]
```

Constant representing the input authority mask.

6.21.2.3 NONE

```
const int NONE = 0 [static]
```

Constant representing no authority.

6.21.2.4 PROXY

```
const int PROXY = 1 << 2 [static]
```

Constant representing the proxy authority mask.

6.21.2.5 STATE

```
const int STATE = 1 << 0 [static]
```

Constant representing the state authority mask.

6.22 Behaviour Class Reference

Alternative base class to Unity's `MonoBehaviour`. This allows for components that work both in Unity, as well as the Photon relays.

Inherits `MonoBehaviour`, `ILogSource`, and `ILogDumpable`.

Inherited by [Hitbox](#), [NetworkEvents](#), [NetworkObject](#), [NetworkObjectPrefabData](#), [NetworkRunner](#), and [SimulationBehaviour](#).

Public Member Functions

- T [AddBehaviour< T > \(\)](#)
Wrapper for Unity's GameObject.AddComponent()
- T [GetBehaviour< T > \(\)](#)
Wrapper for Unity's GameObject.GetComponentInChildren()
- bool [TryGetBehaviour< T > \(out T behaviour\)](#)
Wrapper for Unity's GameObject.TryGetComponent()

Static Public Member Functions

- static void [DestroyBehaviour \(Behaviour behaviour\)](#)
Wrapper for Unity's GameObject.Destroy()

6.22.1 Detailed Description

Alternative base class to Unity's MonoBehaviour. This allows for components that work both in Unity, as well as the Photon relays.

6.22.2 Member Function Documentation

6.22.2.1 AddBehaviour< T >()

```
T AddBehaviour< T > ( )
```

Wrapper for Unity's GameObject.AddComponent()

Type Constraints

T: *Behaviour*

6.22.2.2 DestroyBehaviour()

```
static void DestroyBehaviour (
    Behaviour behaviour ) [static]
```

Wrapper for Unity's GameObject.Destroy()

6.22.2.3 GetBehaviour< T >()

```
T GetBehaviour< T > ( )
```

Wrapper for Unity's `GameObject.GetComponentInChildren()`

Type Constraints

T : *Behaviour*

6.22.2.4 TryGetBehaviour< T >()

```
bool TryGetBehaviour< T > (
    out T behaviour )
```

Wrapper for Unity's `GameObject.TryGetComponent()`

Type Constraints

T : *Behaviour*

6.23 BinaryDataAttribute Class Reference

Specifies that the field represents binary data.

Inherits [DrawerPropertyAttribute](#).

6.23.1 Detailed Description

Specifies that the field represents binary data.

6.24 BinUtils Class Reference

Utility class for binary data.

Static Public Member Functions

- static `unsafe` String `BytesToHex` (byte *buffer, int length, int columns=16, string rowSeparator="\n", string columnSeparator=" ")
Converts a buffer of bytes to a hexadecimal string representation.
- static `unsafe` string `BytesToHex` (byte[] buffer, int columns=16)
Converts a byte array to its hexadecimal string representation.
- static string `ByteToHex` (byte value)
Converts a byte value to its hexadecimal string representation.
- static `unsafe` int `HexToBytes` (string str, byte *buffer, int length)
Converts a hexadecimal string to a byte array.
- static `unsafe` (int, int) `HexToInts`(string str)
Converts a hexadecimal string to an array of 32-bit integers.
- static `unsafe` string `WordsToHex` (int *buffer, int length, int columns=4, string rowSeparator="\n", string columnSeparator=" ")
Converts a buffer of 32-bit integers to a hexadecimal string representation.
- static `unsafe` string `WordsToHex` (uint *buffer, int length, int columns=4, string rowSeparator="\n", string columnSeparator=" ")
Converts a buffer of 32-bit unsigned integers to a hexadecimal string representation.

6.24.1 Detailed Description

Utility class for binary data.

6.24.2 Member Function Documentation

6.24.2.1 BytesToHex() [1/2]

```
static unsafe String BytesToHex (
    byte * buffer,
    int length,
    int columns = 16,
    string rowSeparator = "\n",
    string columnSeparator = " ") [static]
```

Converts a buffer of bytes to a hexadecimal string representation.

Parameters

<i>buffer</i>	A pointer to the buffer containing the bytes to convert.
<i>length</i>	The number of bytes in the buffer.
<i>columns</i>	The number of columns to format the output. Default is 16.
<i>rowSeparator</i>	The string to use as a row separator. Default is newline.
<i>columnSeparator</i>	The string to use as a column separator. Default is a space.

Returns

A string representing the hexadecimal values of the bytes in the buffer.

6.24.2.2 BytesToHex() [2/2]

```
static unsafe string BytesToHex (
    byte[] buffer,
    int columns = 16 ) [static]
```

Converts a byte array to its hexadecimal string representation.

Parameters

<i>buffer</i>	The byte array to convert.
<i>columns</i>	The number of columns to format the output. Default is 16.

Returns

A string representing the hexadecimal values of the bytes in the array.

6.24.2.3 ByteToHex()

```
static string ByteToHex (
    byte value ) [static]
```

Converts a byte value to its hexadecimal string representation.

Parameters

<i>value</i>	The byte value to convert.
--------------	----------------------------

Returns

A string representing the hexadecimal value of the byte.

6.24.2.4 HexToBytes()

```
static unsafe int HexToBytes (
    string str,
    byte * buffer,
    int length ) [static]
```

Converts a hexadecimal string to a byte array.

Parameters

<i>str</i>	The hexadecimal string to convert.
<i>buffer</i>	A pointer to the buffer where the bytes will be stored.
<i>length</i>	The maximum number of bytes to store in the buffer.

Returns

The number of characters processed from the input string.

6.24.2.5 unsafe()

```
static unsafe (
    int ,
    int ) [static]
```

Converts a hexadecimal string to an array of 32-bit integers.

Parameters

<i>str</i>	The hexadecimal string to convert.
<i>buffer</i>	A pointer to the buffer where the 32-bit integers will be stored.
<i>length</i>	The maximum number of 32-bit integers to store in the buffer.

Returns

A tuple containing the number of characters processed from the input string and the number of 32-bit integers stored in the buffer.

6.24.2.6 WordsToHex() [1/2]

```
static unsafe string WordsToHex (
    int * buffer,
    int length,
    int columns = 4,
    string rowSeparator = "\n",
    string columnSeparator = " ") [static]
```

Converts a buffer of 32-bit integers to a hexadecimal string representation.

Parameters

<i>buffer</i>	A pointer to the buffer containing the 32-bit integers to convert.
<i>length</i>	The number of 32-bit integers in the buffer.
<i>columns</i>	The number of columns to format the output. Default is 4.
<i>rowSeparator</i>	The string to use as a row separator. Default is newline.
<i>columnSeparator</i>	The string to use as a column separator. Default is a space.

Returns

A string representing the hexadecimal values of the 32-bit integers in the buffer.

6.24.2.7 WordsToHex() [2/2]

```
static unsafe string WordsToHex (
    uint * buffer,
    int length,
    int columns = 4,
    string rowSeparator = "\n",
    string columnSeparator = " ") [static]
```

Converts a buffer of 32-bit unsigned integers to a hexadecimal string representation.

Parameters

<i>buffer</i>	A pointer to the buffer containing the 32-bit unsigned integers to convert.
<i>length</i>	The number of 32-bit unsigned integers in the buffer.
<i>columns</i>	The number of columns to format the output. Default is 4.
<i>rowSeparator</i>	The string to use as a row separator. Default is newline.
<i>columnSeparator</i>	The string to use as a column separator. Default is a space.

Returns

A string representing the hexadecimal values of the 32-bit unsigned integers in the buffer.

6.25 BitSetAttribute Class Reference

Represents an attribute that specifies the number of bits in a bit set.

Inherits [DrawerPropertyAttribute](#).

Public Member Functions

- [BitSetAttribute](#) (int bitCount)

Initializes a new instance of the [BitSetAttribute](#) class with the specified number of bits.

Properties

- int [BitCount](#) [get]

Gets the number of bits in the bit set.

6.25.1 Detailed Description

Represents an attribute that specifies the number of bits in a bit set.

6.25.2 Constructor & Destructor Documentation

6.25.2.1 BitSetAttribute()

```
BitSetAttribute (
    int bitCount )
```

Initializes a new instance of the [BitSetAttribute](#) class with the specified number of bits.

Parameters

<i>bitCount</i>	The number of bits in the bit set.
-----------------	------------------------------------

6.25.3 Property Documentation

6.25.3.1 BitCount

```
int BitCount [get]
```

Gets the number of bits in the bit set.

6.26 CapacityAttribute Class Reference

Capacity Attribute

Inherits Attribute.

Public Member Functions

- [CapacityAttribute](#) (int length)
CapacityAttribute Constructor

Properties

- int [Length](#) [get]
Total Capacity

6.26.1 Detailed Description

Capacity Attribute

6.26.2 Constructor & Destructor Documentation

6.26.2.1 CapacityAttribute()

```
CapacityAttribute (
    int length )
```

[CapacityAttribute](#) Constructor

Parameters

<i>length</i>	Length
---------------	------------------------

6.26.3 Property Documentation

6.26.3.1 Length

```
int Length [get]
```

Total Capacity

6.27 CRC64 Class Reference

Provides methods to compute [CRC64](#) checksums.

Static Public Member Functions

- static unsafe UInt64 [Compute](#) (Byte *data, Int32 length)
Computes the [CRC64](#) checksum for the given data.
- static unsafe UInt64 [Compute](#) (UInt64 crc, Byte *data, Int32 offset, Int32 length)
Computes the [CRC64](#) checksum for the given data with an initial CRC value and offset.

6.27.1 Detailed Description

Provides methods to compute [CRC64](#) checksums.

6.27.2 Member Function Documentation

6.27.2.1 Compute() [1/2]

```
static unsafe UInt64 Compute (  
    Byte * data,  
    Int32 length ) [static]
```

Computes the [CRC64](#) checksum for the given data.

Parameters

<i>data</i>	A pointer to the data to compute the checksum for.
<i>length</i>	The length of the data.

Returns

The computed [CRC64](#) checksum.

6.27.2.2 Compute() [2/2]

```
static unsafe UInt64 Compute (
    UInt64 crc,
    Byte * data,
    Int32 offset,
    Int32 length ) [static]
```

Computes the [CRC64](#) checksum for the given data with an initial CRC value and offset.

Parameters

<i>crc</i>	The initial CRC value.
<i>data</i>	A pointer to the data to compute the checksum for.
<i>offset</i>	The offset in the data to start computing the checksum from.
<i>length</i>	The length of the data.

Returns

The computed [CRC64](#) checksum.

6.28 DecoratingPropertyAttribute Class Reference

A base class for property attributes that decorate other property attributes.

Inherits [PropertyAttribute](#).

Inherited by [ArrayLengthAttribute](#), [DisplayNameAttribute](#), [DolfAttributeBase](#), [FieldEditorButtonAttribute](#), [HideArrayElementLabelAttribute](#), [InlineHelpAttribute](#), [ReadOnlyAttribute](#), [SerializeReferenceTypePickerAttribute](#), and [UnitAttribute](#).

Static Public Attributes

- const int [DefaultOrder](#) = -10000
The default order of the attribute.

Protected Member Functions

- [DecoratingPropertyAttribute](#) ()
Initializes a new instance with the default order.
- [DecoratingPropertyAttribute](#) (int order)
Initializes a new instance with the specified order.

6.28.1 Detailed Description

A base class for property attributes that decorate other property attributes.

6.28.2 Constructor & Destructor Documentation

6.28.2.1 DecoratingPropertyAttribute() [1/2]

```
DecoratingPropertyAttribute ( ) [protected]
```

Initializes a new instance with the default order.

6.28.2.2 DecoratingPropertyAttribute() [2/2]

```
DecoratingPropertyAttribute (  
    int order ) [protected]
```

Initializes a new instance with the specified order.

Parameters

<i>order</i>	
--------------	--

6.28.3 Member Data Documentation

6.28.3.1 DefaultOrder

```
const int DefaultOrder = -10000 [static]
```

The default order of the attribute.

6.29 DefaultForPropertyAttribute Class Reference

Default For Property Attribute

Inherits [PropertyAttribute](#).

Public Member Functions

- [DefaultForPropertyAttribute](#) (string propertyName, int wordOffset, int wordCount)
DefaultForPropertyAttribute Constructor

Properties

- String [PropertyName](#) [get]
Property Name
- int [WordCount](#) [get]
Property Word Count
- int [WordOffset](#) [get]
Property Word Offset

6.29.1 Detailed Description

Default For Property Attribute

For non-serialized properties

6.29.2 Constructor & Destructor Documentation

6.29.2.1 DefaultForPropertyAttribute()

```
DefaultForPropertyAttribute (
    string propertyName,
    int wordOffset,
    int wordCount )
```

[DefaultForPropertyAttribute](#) Constructor

Parameters

<i>propertyName</i>	PropertyName
<i>wordOffset</i>	WordOffset
<i>wordCount</i>	WordCount

6.29.3 Property Documentation

6.29.3.1 PropertyName

```
String PropertyName [get]
```

Property Name

6.29.3.2 WordCount

```
int WordCount [get]
```

Property Word Count

6.29.3.3 WordOffset

```
int WordOffset [get]
```

Property Word Offset

6.30 DisplayAsEnumAttribute Class Reference

Casts an enum or int value in the inspector to specific enum type for rendering of its popup list. Supplying a method name rather than a type allows a property with the type Type to be used to dynamically get the enum type.

Inherits [DrawerPropertyAttribute](#).

Public Member Functions

- [DisplayAsEnumAttribute](#) (string enumTypeMemberName)
Initializes a new instance of the [DisplayAsEnumAttribute](#) class with the specified enum type member name.
- [DisplayAsEnumAttribute](#) (Type enumType)
Initializes a new instance of the [DisplayAsEnumAttribute](#) class with the specified enum type.

Properties

- Type [EnumType](#) [get]
Gets the type of the enum.
- string [EnumTypeMemberName](#) [get]
Gets the name of the member that returns the enum type.

6.30.1 Detailed Description

Casts an enum or int value in the inspector to specific enum type for rendering of its popup list. Supplying a method name rather than a type allows a property with the type `Type` to be used to dynamically get the enum type.

6.30.2 Constructor & Destructor Documentation

6.30.2.1 `DisplayAsEnumAttribute()` [1/2]

```
DisplayAsEnumAttribute (  
    Type enumType )
```

Initializes a new instance of the `DisplayAsEnumAttribute` class with the specified enum type.

Parameters

<code>enumType</code>	The type of the enum.
-----------------------	-----------------------

6.30.2.2 `DisplayAsEnumAttribute()` [2/2]

```
DisplayAsEnumAttribute (  
    string enumTypeMemberName )
```

Initializes a new instance of the `DisplayAsEnumAttribute` class with the specified enum type member name.

Parameters

<code>enumTypeMemberName</code>	The name of the member that returns the enum type.
---------------------------------	--

6.30.3 Property Documentation

6.30.3.1 `EnumType`

```
Type EnumType [get]
```

Gets the type of the enum.

6.30.3.2 EnumTypeMemberName

```
string EnumTypeMemberName [get]
```

Gets the name of the member that returns the enum type.

6.31 DisplayNameAttribute Class Reference

Specifies the display name for a field.

Inherits [DecoratingPropertyAttribute](#).

Public Member Functions

- [DisplayNameAttribute](#) (string name)
Initializes a new instance of the [DisplayNameAttribute](#) class with the specified name.

Public Attributes

- readonly string [Name](#)
Field name to display.

Additional Inherited Members

6.31.1 Detailed Description

Specifies the display name for a field.

6.31.2 Constructor & Destructor Documentation

6.31.2.1 DisplayNameAttribute()

```
DisplayNameAttribute (  
    string name )
```

Initializes a new instance of the [DisplayNameAttribute](#) class with the specified name.

Parameters

<i>name</i>	The display name.
-------------	-------------------

6.31.3 Member Data Documentation

6.31.3.1 Name

readonly string Name

Field name to display.

6.32 DolfAttributeBase Class Reference

Editor attribute for selective editor rendering. Condition member can be a property, field or method (with a return value).

Inherits [DecoratingPropertyAttribute](#).

Inherited by [DrawIfAttribute](#), [ErrorIfAttribute](#), and [WarnIfAttribute](#).

Public Attributes

- double [_doubleValue](#)
The double value to compare against.
- bool [_isDouble](#)
Is the value to compare against a double?
- long [_longValue](#)
The long value to compare against.
- [CompareOperator Compare](#)
The comparison operator to use.
- string [ConditionMember](#)
Condition member to evaluate.
- bool [ErrorOnConditionMemberNotFound](#) = true
If `true`, an error will be thrown if the condition member is not found.

Protected Member Functions

- [DolfAttributeBase](#) (string conditionMember, bool compareToValue, [CompareOperator](#) compare)
Initializes a new instance with a boolean value to compare against.
- [DolfAttributeBase](#) (string conditionMember, double compareToValue, [CompareOperator](#) compare)
Initializes a new instance with a double value to compare against.
- [DolfAttributeBase](#) (string conditionMember, long compareToValue, [CompareOperator](#) compare)
Initializes a new instance with a long value to compare against.

Additional Inherited Members

6.32.1 Detailed Description

Editor attribute for selective editor rendering. Condition member can be a property, field or method (with a return value).

Value of condition method is converted to a long or a double. `null = 0`, `false = 0`, `true = 1`, `Unity Object = InstanceId`

6.32.2 Constructor & Destructor Documentation

6.32.2.1 DolfAttributeBase() [1/3]

```
DoIfAttributeBase (
    string conditionMember,
    double compareToValue,
    CompareOperator compare ) [protected]
```

Initializes a new instance with a double value to compare against.

Parameters

<i>conditionMember</i>	
<i>compareToValue</i>	
<i>compare</i>	

6.32.2.2 DolfAttributeBase() [2/3]

```
DoIfAttributeBase (
    string conditionMember,
    long compareToValue,
    CompareOperator compare ) [protected]
```

Initializes a new instance with a long value to compare against.

Parameters

<i>conditionMember</i>	
<i>compareToValue</i>	
<i>compare</i>	

6.32.2.3 DofAttributeBase() [3/3]

```
DofAttributeBase (  
    string conditionMember,  
    bool compareToValue,  
    CompareOperator compare ) [protected]
```

Initializes a new instance with a boolean value to compare against.

Parameters

<i>conditionMember</i>	
<i>compareToValue</i>	
<i>compare</i>	

6.32.3 Member Data Documentation

6.32.3.1 _doubleValue

```
double _doubleValue
```

The double value to compare against.

6.32.3.2 _isDouble

```
bool _isDouble
```

Is the value to compare against a double?

6.32.3.3 _longValue

```
long _longValue
```

The long value to compare against.

6.32.3.4 Compare

```
CompareOperator Compare
```

The comparison operator to use.

6.32.3.5 ConditionMember

```
string ConditionMember
```

Condition member to evaluate.

6.32.3.6 ErrorOnConditionMemberNotFound

```
bool ErrorOnConditionMemberNotFound = true
```

If `true`, an error will be thrown if the condition member is not found.

6.33 DrawerPropertyAttribute Class Reference

A base class for property attributes that are used to draw properties in the inspector.

Inherits [PropertyAttribute](#).

Inherited by [AssemblyNameAttribute](#), [BinaryDataAttribute](#), [BitSetAttribute](#), [DisplayAsEnumAttribute](#), [DrawInlineAttribute](#), [ExpandableEnumAttribute](#), [LayerAttribute](#), [LayerMatrixAttribute](#), [MaxStringByteCountAttribute](#), [RangeExAttribute](#), [ScenePathAttribute](#), [ToggleLeftAttribute](#), [UnityAssetGuidAttribute](#), and [UnityResourcePathAttribute](#).

6.33.1 Detailed Description

A base class for property attributes that are used to draw properties in the inspector.

6.34 DrawIfAttribute Class Reference

Editor attribute for selectively drawing/hiding fields. Condition member can be a property, field or method (with a return value).

Inherits [DofAttributeBase](#).

Public Member Functions

- [DrawIfAttribute](#) (string conditionMember)
Initializes a new instance that will hide the field if the condition member is not equal to zero.
- [DrawIfAttribute](#) (string conditionMember, bool compareToValue, [CompareOperator](#) compare=[CompareOperator.Equal](#), [DrawIfMode](#) mode=[DrawIfMode.ReadOnly](#))
- [DrawIfAttribute](#) (string conditionMember, double compareToValue, [CompareOperator](#) compare=[CompareOperator.Equal](#), [DrawIfMode](#) mode=[DrawIfMode.ReadOnly](#))
- [DrawIfAttribute](#) (string conditionMember, long compareToValue, [CompareOperator](#) compare=[CompareOperator.Equal](#), [DrawIfMode](#) mode=[DrawIfMode.ReadOnly](#))

Public Attributes

- [DrawIfMode Mode](#)

Instructs the attribute completely hide the field if not draw, rather than the default of just disabling it.

Properties

- bool? [Hide](#) [get, set]

Should the field be hidden if the condition is not met?

Additional Inherited Members

6.34.1 Detailed Description

Editor attribute for selectively drawing/hiding fields. Condition member can be a property, field or method (with a return value).

Value of condition method is converted to a long. `null = 0`, `false = 0`, `true = 1`, `Unity Object = InstanceId`

6.34.2 Constructor & Destructor Documentation

6.34.2.1 DrawIfAttribute()

```
DrawIfAttribute (
    string conditionMember )
```

Initializes a new instance that will hide the field if the condition member is not equal to zero.

Parameters

<code>conditionMember</code>	
------------------------------	--

6.34.3 Member Data Documentation

6.34.3.1 Mode

```
DrawIfMode Mode
```

Instructs the attribute completely hide the field if not draw, rather than the default of just disabling it.

6.34.4 Property Documentation

6.34.4.1 Hide

```
bool? Hide [get], [set]
```

Should the field be hidden if the condition is not met?

6.35 DrawInlineAttribute Class Reference

Specifies that a field should be drawn inline in the inspector.

Inherits [DrawerPropertyAttribute](#).

6.35.1 Detailed Description

Specifies that a field should be drawn inline in the inspector.

6.36 DynamicHeap Struct Reference

A dynamic heap for allocating and tracking unmanaged objects.

Classes

- class [Ignore](#)
Ignore this field when scanning for pointers.

Public Types

- enum class **ObjectFlags** : byte

Public Member Functions

- delegate void [CollectGarbageDelegate](#) ([DynamicHeap](#) *heap, void **dynamicRoots, int dynamicRoots↔
Length)
Collect garbage delegate

Static Public Member Functions

- static void **AllocateBlock** ([DynamicHeap](#) *heap)
- static byte * **AllocateInternal** ([DynamicHeap](#) *heap, int size, out byte block)
- static Page * **AllocatePage** ([DynamicHeap](#) *heap)
- static Page * **AllocatePage_Internal** ([DynamicHeap](#) *heap, bool mustSucceed)
- static int **BitScan** (uint v)
- static int **BlocksWithAvailablePages** ([DynamicHeap](#) *heap)
- static void **CollectGarbage** ([DynamicHeap](#) *heap, void **dynamicRoots, int dynamicRootsLength)
 - Collect garbage*
- static void **Destroy** (Block *block)
- static void **ExpandStack** ([DynamicHeap](#) *heap)
- static void **Free** ([DynamicHeap](#) *heap, void *ptr)
 - Free up an object*
- static void **FreeInternal** ([DynamicHeap](#) *heap, void *ptr, Object objData)
- static int **GetBin** (int size)
- static Bin * **GetBinByIndex** ([DynamicHeap](#) *heap, int binIndex)
- static int **GetBinIndexForSize** ([DynamicHeap](#) *heap, int size)
- static Page * **GetPageForPtr** ([DynamicHeap](#) *heap, Block *block, void *ptr)
- static int **GetPageOffset** (Page *page, ObjectFree *obj)
- static ushort **GetTypeOffset**< T > ()
- static void **InitObj** ([DynamicHeap](#) *heap, Object *obj, ushort type, ushort array, byte block)
- static void **InitRoot** ([DynamicHeap](#) *heap, Object *obj)
- static bool **IsPtrInBlock** ([DynamicHeap](#) *heap, Block *block, void *p)
- static ushort **NextGen** ([DynamicHeap](#) *heap)
- static int **ObjectsFreeCount** (Page *p)
- static int **PagesWithAvailableObjectsInBin** (Bin *bin)
- static ObjectFree * **ResolvePageOffset** (Page *page, int offset)
- static T * **SetForcedAlive**< T > (T *ptr)
 - Mark an object with ObjectFlags.ForceAlive*
- static void **ThrowHeapCorrupted** ()
- static byte * **TryAllocateFromPage** ([DynamicHeap](#) *heap, Page *page, int size, out byte block)
- static int **WordCount** (int size)

Public Attributes

- Bin * **_bins**
- Block ** **_blocks**
- BlockList **_blocksFreePages**
- int **_blocksUsed**
- Config **_config**
- int **_gcBlock**
- int **_gcBlockPage**
- ushort **_gcGen**
- Phase **_gcPhase**
- Object ** **_gcStack**
- int **_gcStackCapacity**
- int **_gcStackCount**
- int **_memoryAllocated**
- int **_objectsAllocated**
- Object ** **_rootList**
- int **_rootListCapacity**
- int **_rootListCount**
- int * **_typeMap**
- int **_typeMapLength**
- int * **_typeMapStrides**

Static Public Attributes

- static byte[] **_debruijnTable**
- static Dictionary< Type, TypeData > **_types** = null
- static Dictionary< ushort, TypeData > **_typesByOffset** = null

6.36.1 Detailed Description

A dynamic heap for allocating and tracking unmanaged objects.

6.36.2 Member Function Documentation

6.36.2.1 CollectGarbage()

```
static void CollectGarbage (
    DynamicHeap * heap,
    void ** dynamicRoots,
    int dynamicRootsLength ) [static]
```

Collect garbage

Parameters

<i>heap</i>	Dynamic heap to collect from
<i>dynamicRoots</i>	Dynamic roots
<i>dynamicRootsLength</i>	Dynamic roots length

6.36.2.2 CollectGarbageDelegate()

```
delegate void CollectGarbageDelegate (
    DynamicHeap * heap,
    void ** dynamicRoots,
    int dynamicRootsLength )
```

Collect garbage delegate

6.36.2.3 Free()

```
static void Free (
    DynamicHeap * heap,
    void * ptr ) [static]
```

Free up an object

Parameters

<i>heap</i>	Heap to free from
<i>ptr</i>	Pointer to object

Exceptions

<i>InvalidOperationException</i>	Thrown if <i>ptr</i> is not a tracked object
----------------------------------	--

6.36.2.4 SetForcedAlive< T >()

```
static T* SetForcedAlive< T > (
    T * ptr ) [static]
```

Mark an object with ObjectFlags.ForceAlive

Parameters

<i>ptr</i>	Pointer Object to mark
------------	------------------------

Template Parameters

<i>T</i>	Type of object
----------	----------------

Returns

Pointer to object

Type Constraints

T*: *unmanaged

6.36.3 Member Data Documentation**6.36.3.1 _debruijnTable**

```
byte [] _debruijnTable [static]
```

Initial value:

```
= {
    0, 9, 1, 10, 13, 21, 2, 29, 11, 14, 16, 18, 22, 25, 3, 30,
    8, 12, 20, 28, 15, 17, 24, 7, 19, 27, 23, 6, 26, 5, 4, 31
}
```


6.37 DynamicHeap.Ignore Class Reference

[Ignore](#) this field when scanning for pointers.

Inherits Attribute.

6.37.1 Detailed Description

[Ignore](#) this field when scanning for pointers.

6.38 DynamicHeapInstance Class Reference

Dynamic heap instance.

Public Member Functions

- void * [Allocate](#) (int size)
Allocate a pointer.
- void * [AllocateArray](#)< T > (int length)
Allocate a pointer array.
- void * [AllocateArrayPointers](#)< T > (int length)
Allocate an array of pointers.
- void * [AllocateTracked](#)< T > (bool root=false)
Allocate a tracked pointer.
- void * [AllocateTrackedArray](#)< T > (int length, bool root=false)
Allocate a tracked pointer array.
- void * [AllocateTrackedArrayPointers](#)< T > (int length, bool root=false)
Allocate a tracked array of pointers.
- [DynamicHeapInstance](#) (params Type[] types)
Create a dynamic heap instance.
- void [Free](#) (void *ptr)
Free a pointer.

6.38.1 Detailed Description

Dynamic heap instance.

6.38.2 Constructor & Destructor Documentation

6.38.2.1 DynamicHeapInstance()

```
DynamicHeapInstance (
    params Type[] types )
```

Create a dynamic heap instance.

Parameters

<i>types</i>	Types to allocate.
--------------	--------------------

6.38.3 Member Function Documentation

6.38.3.1 Allocate()

```
void* Allocate (
    int size )
```

Allocate a pointer.

Parameters

<i>size</i>	Size to allocate.
-------------	-------------------

Returns

Pointer to allocated memory.

6.38.3.2 AllocateArray< T >()

```
void* AllocateArray< T > (
    int length )
```

Allocate a pointer array.

Parameters

<i>length</i>	Length of array.
---------------	------------------

Template Parameters

<i>T</i>	Type of array.
----------	----------------

Returns

Pointer to allocated memory.

Type Constraints

T* : *unmanaged

6.38.3.3 AllocateArrayPointers< T >()

```
void* AllocateArrayPointers< T > (
    int length )
```

Allocate an array of pointers.

Parameters

<i>length</i>	Length of array.
---------------	------------------

Template Parameters

<i>T</i>	Type of array.
----------	----------------

Returns

Pointer to allocated memory.

Type Constraints

T* : *unmanaged

6.38.3.4 AllocateTracked< T >()

```
void* AllocateTracked< T > (
    bool root = false )
```

Allocate a tracked pointer.

Parameters

<i>root</i>	Signal if the pointer is a root.
-------------	----------------------------------

Template Parameters

<i>T</i>	Type of pointer.
----------	------------------

Returns

Pointer to allocated memory.

Type Constraints

T* : *unmanaged

6.38.3.5 AllocateTrackedArray< T >()

```
void* AllocateTrackedArray< T > (
    int length,
    bool root = false )
```

Allocate a tracked pointer array.

Parameters

<i>length</i>	Length of array.
<i>root</i>	Signal if the pointer is a root.

Template Parameters

<i>T</i>	Type of array.
----------	----------------

Returns

Pointer to allocated memory.

Type Constraints

T* : *unmanaged

6.38.3.6 AllocateTrackedArrayPointers< T >()

```
void* AllocateTrackedArrayPointers< T > (
    int length,
    bool root = false )
```

Allocate a tracked array of pointers.

Parameters

<i>length</i>	Length of array.
<i>root</i>	Signal if the pointer is a root.

Template Parameters

<i>T</i>	Type of array.
----------	----------------

Returns

Pointer to allocated memory.

Type Constraints

T: *unmanaged*

6.38.3.7 Free()

```
void Free (
    void * ptr )
```

Free a pointer.

Parameters

<i>ptr</i>	Pointer to free.
------------	------------------

6.39 EditorButtonAttribute Class Reference

Specifies that a method should be displayed as a button in the Unity editor.

Inherits Attribute.

Public Member Functions

- [EditorButtonAttribute](#) ([EditorButtonVisibility](#) visibility=[EditorButtonVisibility.Always](#), int priority=0, bool dirty↔Object=false)
Initializes a new instance of the [EditorButtonAttribute](#) class with the specified visibility, priority, and dirty object flag.
- [EditorButtonAttribute](#) (string label, [EditorButtonVisibility](#) visibility=[EditorButtonVisibility.Always](#), int priority=0, bool dirtyObject=false)
Initializes a new instance of the [EditorButtonAttribute](#) class with the specified label, visibility, priority, and dirty object flag.

Public Attributes

- bool [AllowMultipleTargets](#)
Determines whether multiple targets are supported.
- bool [DirtyObject](#)
Determines whether the object should be marked as dirty after clicking the button.
- string [Label](#)
The label text to display on the button.
- int [Priority](#)
The priority of the button. Buttons with higher priority are displayed first.
- [EditorButtonVisibility](#) [Visibility](#)
The visibility of the button in the Unity editor.

6.39.1 Detailed Description

Specifies that a method should be displayed as a button in the Unity editor.

6.39.2 Constructor & Destructor Documentation

6.39.2.1 EditorButtonAttribute() [1/2]

```
EditorButtonAttribute (
    string label,
    EditorButtonVisibility visibility = EditorButtonVisibility.Always,
    int priority = 0,
    bool dirtyObject = false )
```

Initializes a new instance of the [EditorButtonAttribute](#) class with the specified label, visibility, priority, and dirty object flag.

Parameters

<i>label</i>	The label text to display on the button.
<i>visibility</i>	The visibility of the button in the Unity editor.
<i>priority</i>	The priority of the button. Buttons with higher priority are displayed first.
<i>dirtyObject</i>	Determines whether the object should be marked as dirty after clicking the button.

6.39.2.2 EditorButtonAttribute() [2/2]

```
EditorButtonAttribute (
    EditorButtonVisibility visibility = EditorButtonVisibility.Always,
    int priority = 0,
    bool dirtyObject = false )
```

Initializes a new instance of the [EditorButtonAttribute](#) class with the specified visibility, priority, and dirty object flag.

Parameters

<i>visibility</i>	The visibility of the button in the Unity editor.
<i>priority</i>	The priority of the button. Buttons with higher priority are displayed first.
<i>dirtyObject</i>	Determines whether the object should be marked as dirty after clicking the button.

6.39.3 Member Data Documentation

6.39.3.1 AllowMultipleTargets

```
bool AllowMultipleTargets
```

Determines whether multiple targets are supported.

6.39.3.2 DirtyObject

```
bool DirtyObject
```

Determines whether the object should be marked as dirty after clicking the button.

6.39.3.3 Label

```
string Label
```

The label text to display on the button.

6.39.3.4 Priority

```
int Priority
```

The priority of the button. Buttons with higher priority are displayed first.

6.39.3.5 Visibility

```
EditorButtonVisibility Visibility
```

The visibility of the button in the Unity editor.

6.40 DataEncryptor Class Reference

Responsible for encrypting and decrypting data buffers

Inherits [IDataEncryption](#).

Public Member Functions

- bool [ComputeHash](#) (byte *buffer, ref int bufferLength, int capacity)
Compute the Buffer hash and append it to the buffer itself
- bool [DecryptData](#) (byte *buffer, ref int bufferLength, int capacity)
Decrypt data in place and update it's length.
- void [Dispose](#) ()
Dispose of the [DataEncryptor](#)
- bool [EncryptData](#) (byte *buffer, ref int bufferLength, int capacity)
Encrypts the data in the provided buffer.
- byte[] [GenerateKey](#) ()
Generate the key used used by the encryption implementation
- void [Setup](#) (byte[] key)
Setup the encryption implementation with the right key
- bool [VerifyHash](#) (byte *buffer, ref int bufferLength, int capacity)
Verify the buffer hash that was appended to the buffer

6.40.1 Detailed Description

Responsible for encrypting and decrypting data buffers

6.40.2 Member Function Documentation

6.40.2.1 Dispose()

```
void Dispose ( )
```

Dispose of the [DataEncryptor](#)

6.40.2.2 EncryptData()

```
bool EncryptData (
    byte * buffer,
    ref int bufferLength,
    int capacity )
```

Encrypts the data in the provided buffer.

Parameters

<i>buffer</i>	The buffer containing the data to be encrypted.
<i>bufferLength</i>	The length of the data in the buffer.
<i>capacity</i>	The total capacity of the buffer.

Returns

Returns true if the encryption was successful, false otherwise.

Exceptions

<i>InvalidOperationException</i>	Thrown when the encryption provider is not initialized.
<i>ArgumentException</i>	Thrown when the original buffer cannot hold the encrypted data.

Implements [IDataEncryption](#).

6.41 IDataEncryption Interface Reference

Interface for classes that manage the encryption/decryption of byte arrays

Inherits [IDisposable](#).

Inherited by [DataEncryptor](#).

Public Member Functions

- bool [ComputeHash](#) (byte *buffer, ref int bufferLength, int capacity)
Compute the Buffer hash and append it to the buffer itself
- bool [DecryptData](#) (byte *buffer, ref int bufferLength, int capacity)
Decrypt data in place and update it's length.
- bool [EncryptData](#) (byte *buffer, ref int bufferLength, int capacity)
Encrypt data in place and update it's length.
- byte[] [GenerateKey](#) ()
Generate the key used used by the encryption implementation
- void [Setup](#) (byte[] key)
Setup the encryption implementation with the right key
- bool [VerifyHash](#) (byte *buffer, ref int bufferLength, int capacity)
Verify the buffer hash that was appended to the buffer

6.41.1 Detailed Description

Interface for classes that manage the encryption/decryption of byte arrays

6.41.2 Member Function Documentation

6.41.2.1 ComputeHash()

```
bool ComputeHash (
    byte * buffer,
    ref int bufferLength,
    int capacity )
```

Compute the Buffer hash and append it to the buffer itself

Parameters

<i>buffer</i>	Data to compute the hash
<i>bufferLength</i>	Length of the data to hash
<i>capacity</i>	Buffer total capacity

Returns

True if the hash was properly computed, false otherwise

Implemented in [DataEncryptor](#).

6.41.2.2 DecryptData()

```
bool DecryptData (
    byte * buffer,
    ref int bufferLength,
    int capacity )
```

Decrypt data in place and update it's length.

Parameters

<i>buffer</i>	Data to decrypt
<i>bufferLength</i>	Length of the data to decrypt
<i>capacity</i>	Buffer total capacity

Returns

True if the decryption was completed, false otherwise

Implemented in [DataEncryptor](#).

6.41.2.3 EncryptData()

```
bool EncryptData (
    byte * buffer,
    ref int bufferLength,
    int capacity )
```

Encrypt data in place and update it's length.

Parameters

<i>buffer</i>	Data to encrypt
<i>bufferLength</i>	Length of the data to encrypt
<i>capacity</i>	Buffer total capacity

Returns

True if the encryption was completed, false otherwise

Implemented in [DataEncryptor](#).

6.41.2.4 GenerateKey()

```
byte [] GenerateKey ( )
```

Generate the key used used by the encryption implementation

Returns

Key used to setup the encryption implementation

Implemented in [DataEncryptor](#).

6.41.2.5 Setup()

```
void Setup (
    byte[] key )
```

Setup the encryption implementation with the right key

Implemented in [DataEncryptor](#).

6.41.2.6 VerifyHash()

```
bool VerifyHash (
    byte * buffer,
    ref int bufferLength,
    int capacity )
```

Verify the buffer hash that was appended to the buffer

Parameters

<i>buffer</i>	Buffer to check the hash
<i>bufferLength</i>	Length of the data to hash
<i>capacity</i>	Buffer total capacity

Returns

True if the hash was properly verified, false otherwise

Implemented in [DataEncryptor](#).

6.42 EncryptionConfig Class Reference

Configuration for the [Encryption](#) Feature

Public Attributes

- bool [EnableEncryption](#)
Enabled the [Encryption](#) Feature

6.42.1 Detailed Description

Configuration for the [Encryption](#) Feature

6.42.2 Member Data Documentation

6.42.2.1 EnableEncryption

```
bool EnableEncryption
```

Enabled the [Encryption](#) Feature

6.43 EngineProfiler Class Reference

Provides a set of methods to profile the engine.

Static Public Member Functions

- static void [Begin](#) (string sample)
Begins a profiling sample with the specified name.
- static void [End](#) ()
Ends the current profiling sample.
- static void [InputQueue](#) (int value)
Invokes the input queue callback with the specified value.
- static void [InputRecvDelta](#) (float value)
Invokes the input receive delta callback with the specified value.
- static void [InputRecvDeltaDeviation](#) (float value)
Invokes the input receive delta deviation callback with the specified value.
- static void [InputSize](#) (int value)
Invokes the input size callback with the specified value.
- static void [InterpolationOffset](#) (float value)
Invokes the interpolation offset callback with the specified value.
- static void [InterpolationOffsetDeviation](#) (float value)
Invokes the interpolation offset deviation callback with the specified value.
- static void [InterpolationSpeed](#) (float value)
Invokes the interpolation speed callback with the specified value.
- static void [Resimulations](#) (int value)
Invokes the resimulations callback with the specified value.
- static void [RoundTripTime](#) (float value)
Invokes the round trip time callback with the specified value.
- static void [RpcIn](#) (int value)
Invokes the RPC in callback with the specified value.
- static void [RpcOut](#) (int value)
Invokes the RPC out callback with the specified value.
- static void [SimulationOffset](#) (float value)
Invokes the simulation offset callback with the specified value.
- static void [SimulationOffsetDeviation](#) (float value)
Invokes the simulation offset deviation callback with the specified value.
- static void [SimulationSpeed](#) (float value)
Invokes the simulation speed callback with the specified value.
- static void [StateRecvDelta](#) (float value)
Invokes the state receive delta callback with the specified value.
- static void [StateRecvDeltaDeviation](#) (float value)
Invokes the state receive delta deviation callback with the specified value.
- static void [WorldSnapshotSize](#) (int value)
Invokes the world snapshot size callback with the specified value.

Static Public Attributes

- static Action< int > [InputQueueCallback](#)
Callback for input queue profiling.
- static Action< float > [InputRecvDeltaCallback](#)
Callback for input receive delta profiling.
- static Action< float > [InputRecvDeltaDeviationCallback](#)
Callback for input receive delta deviation profiling.
- static Action< int > [InputSizeCallback](#)

- Callback for input size profiling.*

 - static Action< float > [InterpolationOffsetCallback](#)

Callback for interpolation offset profiling.
- static Action< float > [InterpolationOffsetDeviationCallback](#)

Callback for interpolation offset deviation profiling.
- static Action< float > [InterpolationSpeedCallback](#)

Callback for interpolation speed profiling.
- static Action< int > [ResimulationsCallback](#)

Callback for resimulations profiling.
- static Action< float > [RoundTripTimeCallback](#)

Callback for round trip time profiling.
- static Action< int > [RpcInCallback](#)

Callback for RPC in profiling.
- static Action< int > [RpcOutCallback](#)

Callback for RPC out profiling.
- static Action< float > [SimulationOffsetCallback](#)

Callback for simulation offset profiling.
- static Action< float > [SimulationOffsetDeviationCallback](#)

Callback for simulation offset deviation profiling.
- static Action< float > [SimulationSpeedCallback](#)

Callback for simulation speed profiling.
- static Action< float > [StateRecvDeltaCallback](#)

Callback for state receive delta profiling.
- static Action< float > [StateRecvDeltaDeviationCallback](#)

Callback for state receive delta deviation profiling.
- static Action< int > [WorldSnapshotSizeCallback](#)

Callback for world snapshot size profiling.

6.43.1 Detailed Description

Provides a set of methods to profile the engine.

6.43.2 Member Function Documentation

6.43.2.1 Begin()

```
static void Begin (
    string sample ) [static]
```

Begins a profiling sample with the specified name.

Parameters

<i>sample</i>	The name of the profiling sample.
---------------	-----------------------------------

6.43.2.2 End()

```
static void End ( ) [static]
```

Ends the current profiling sample.

6.43.2.3 InputQueue()

```
static void InputQueue (
    int value ) [static]
```

Invokes the input queue callback with the specified value.

Parameters

<i>value</i>	The input queue value.
--------------	------------------------

6.43.2.4 InputRecvDelta()

```
static void InputRecvDelta (
    float value ) [static]
```

Invokes the input receive delta callback with the specified value.

Parameters

<i>value</i>	The input receive delta value.
--------------	--------------------------------

6.43.2.5 InputRecvDeltaDeviation()

```
static void InputRecvDeltaDeviation (
    float value ) [static]
```

Invokes the input receive delta deviation callback with the specified value.

Parameters

<i>value</i>	The input receive delta deviation value.
--------------	--

6.43.2.6 InputSize()

```
static void InputSize (  
    int value ) [static]
```

Invokes the input size callback with the specified value.

Parameters

<i>value</i>	The input size value.
--------------	-----------------------

6.43.2.7 InterpolationOffset()

```
static void InterpolationOffset (  
    float value ) [static]
```

Invokes the interpolation offset callback with the specified value.

Parameters

<i>value</i>	The interpolation offset value.
--------------	---------------------------------

6.43.2.8 InterpolationOffsetDeviation()

```
static void InterpolationOffsetDeviation (  
    float value ) [static]
```

Invokes the interpolation offset deviation callback with the specified value.

Parameters

<i>value</i>	The interpolation offset deviation value.
--------------	---

6.43.2.9 InterpolationSpeed()

```
static void InterpolationSpeed (  
    float value ) [static]
```

Invokes the interpolation speed callback with the specified value.

Parameters

<i>value</i>	The interpolation speed value.
--------------	--------------------------------

6.43.2.10 Resimulations()

```
static void Resimulations (  
    int value ) [static]
```

Invokes the resimulations callback with the specified value.

Parameters

<i>value</i>	The resimulations value.
--------------	--------------------------

6.43.2.11 RoundTripTime()

```
static void RoundTripTime (  
    float value ) [static]
```

Invokes the round trip time callback with the specified value.

Parameters

<i>value</i>	The round trip time value.
--------------	----------------------------

6.43.2.12 RpcIn()

```
static void RpcIn (  
    int value ) [static]
```

Invokes the RPC in callback with the specified value.

Parameters

<i>value</i>	The RPC in value.
--------------	-------------------

6.43.2.13 RpcOut()

```
static void RpcOut (
    int value ) [static]
```

Invokes the RPC out callback with the specified value.

Parameters

<i>value</i>	The RPC out value.
--------------	--------------------

6.43.2.14 SimulationOffset()

```
static void SimulationOffset (
    float value ) [static]
```

Invokes the simulation offset callback with the specified value.

Parameters

<i>value</i>	The simulation offset value.
--------------	------------------------------

6.43.2.15 SimulationOffsetDeviation()

```
static void SimulationOffsetDeviation (
    float value ) [static]
```

Invokes the simulation offset deviation callback with the specified value.

Parameters

<i>value</i>	The simulation offset deviation value.
--------------	--

6.43.2.16 SimulationSpeed()

```
static void SimulationSpeed (
    float value ) [static]
```

Invokes the simulation speed callback with the specified value.

Parameters

<i>value</i>	The simulation speed value.
--------------	-----------------------------

6.43.2.17 StateRecvDelta()

```
static void StateRecvDelta (  
    float value ) [static]
```

Invokes the state receive delta callback with the specified value.

Parameters

<i>value</i>	The state receive delta value.
--------------	--------------------------------

6.43.2.18 StateRecvDeltaDeviation()

```
static void StateRecvDeltaDeviation (  
    float value ) [static]
```

Invokes the state receive delta deviation callback with the specified value.

Parameters

<i>value</i>	The state receive delta deviation value.
--------------	--

6.43.2.19 WorldSnapshotSize()

```
static void WorldSnapshotSize (  
    int value ) [static]
```

Invokes the world snapshot size callback with the specified value.

Parameters

<i>value</i>	The world snapshot size value.
--------------	--------------------------------

6.43.3 Member Data Documentation

6.43.3.1 InputQueueCallback

Action<int> InputQueueCallback [static]

Callback for input queue profiling.

6.43.3.2 InputRecvDeltaCallback

Action<float> InputRecvDeltaCallback [static]

Callback for input receive delta profiling.

6.43.3.3 InputRecvDeltaDeviationCallback

Action<float> InputRecvDeltaDeviationCallback [static]

Callback for input receive delta deviation profiling.

6.43.3.4 InputSizeCallback

Action<int> InputSizeCallback [static]

Callback for input size profiling.

6.43.3.5 InterpolationOffsetCallback

Action<float> InterpolationOffsetCallback [static]

Callback for interpolation offset profiling.

6.43.3.6 InterpolationOffsetDeviationCallback

Action<float> InterpolationOffsetDeviationCallback [static]

Callback for interpolation offset deviation profiling.

6.43.3.7 InterpolationSpeedCallback

Action<float> InterpolationSpeedCallback [static]

Callback for interpolation speed profiling.

6.43.3.8 ResimulationsCallback

Action<int> ResimulationsCallback [static]

Callback for resimulations profiling.

6.43.3.9 RoundTripTimeCallback

Action<float> RoundTripTimeCallback [static]

Callback for round trip time profiling.

6.43.3.10 RpcInCallback

Action<int> RpcInCallback [static]

Callback for RPC in profiling.

6.43.3.11 RpcOutCallback

Action<int> RpcOutCallback [static]

Callback for RPC out profiling.

6.43.3.12 SimulationOffsetCallback

Action<float> SimulationOffsetCallback [static]

Callback for simulation offset profiling.

6.43.3.13 SimulationOffsetDeviationCallback

Action<float> SimulationOffsetDeviationCallback [static]

Callback for simulation offset deviation profiling.

6.43.3.14 SimulationSpeedCallback

Action<float> SimulationSpeedCallback [static]

Callback for simulation speed profiling.

6.43.3.15 StateRecvDeltaCallback

Action<float> StateRecvDeltaCallback [static]

Callback for state receive delta profiling.

6.43.3.16 StateRecvDeltaDeviationCallback

Action<float> StateRecvDeltaDeviationCallback [static]

Callback for state receive delta deviation profiling.

6.43.3.17 WorldSnapshotSizeCallback

Action<int> WorldSnapshotSizeCallback [static]

Callback for world snapshot size profiling.

6.44 ErrorIfAttribute Class Reference

Editor attribute for adding notices to fields if the condition member evaluates as `true`. Condition member can be a property, field or method (with a return value).

Inherits [DofAttributeBase](#).

Public Member Functions

- [ErrorIfAttribute](#) (string conditionMember, bool compareToValue, string message, [CompareOperator](#) compare=[CompareOperator.Equal](#))
- [ErrorIfAttribute](#) (string conditionMember, double compareToValue, string message, [CompareOperator](#) compare=[CompareOperator.Equal](#))
- [ErrorIfAttribute](#) (string conditionMember, long compareToValue, string message, [CompareOperator](#) compare=[CompareOperator.Equal](#))

Public Attributes

- bool [AsBox](#)
Should the error be shown as a box?
- string [Message](#)
The default error text, when an error is shown.

Additional Inherited Members

6.44.1 Detailed Description

Editor attribute for adding notices to fields if the condition member evaluates as `true`. Condition member can be a property, field or method (with a return value).

6.44.2 Member Data Documentation

6.44.2.1 AsBox

```
bool AsBox
```

Should the error be shown as a box?

6.44.2.2 Message

```
string Message
```

The default error text, when an error is shown.

6.45 ExpandableEnumAttribute Class Reference

Editor attribute that shows an enum as an expandable list of options in the inspector.

Inherits [DrawerPropertyAttribute](#).

Properties

- bool `AlwaysExpanded` = false [get, set]
Always expand the enum in the inspector (no foldout)
- bool `ShowFlagsButtons` = true [get, set]
Show the enum flags as buttons in the inspector.
- bool `ShowInlineHelp` = false [get, set]
Show inline help for enum values in the inspector.

6.45.1 Detailed Description

Editor attribute that shows an enum as an expandable list of options in the inspector.

6.45.2 Property Documentation

6.45.2.1 AlwaysExpanded

```
bool AlwaysExpanded = false [get], [set]
```

Always expand the enum in the inspector (no foldout)

6.45.2.2 ShowFlagsButtons

```
bool ShowFlagsButtons = true [get], [set]
```

Show the enum flags as buttons in the inspector.

6.45.2.3 ShowInlineHelp

```
bool ShowInlineHelp = false [get], [set]
```

Show inline help for enum values in the inspector.

6.46 FieldEditorButtonAttribute Class Reference

Editor attribute to add a button that invokes a custom method in the inspector.

Inherits [DecoratingPropertyAttribute](#).

Public Member Functions

- [FieldEditorButtonAttribute](#) (string label, string targetMethod)
Initializes a new instance class with the specified label and target method.

Public Attributes

- bool [AllowMultipleTargets](#)
Is it allowed to select multiple targets for the button?
- string [Label](#)
Button label.
- string [TargetMethod](#)
The method to invoke when the button is clicked.

Additional Inherited Members

6.46.1 Detailed Description

Editor attribute to add a button that invokes a custom method in the inspector.

6.46.2 Constructor & Destructor Documentation

6.46.2.1 FieldEditorButtonAttribute()

```
FieldEditorButtonAttribute (
    string label,
    string targetMethod )
```

Initializes a new instance class with the specified label and target method.

Parameters

<i>label</i>	The label of the button
<i>targetMethod</i>	The method to invoke when the button is clicked

6.46.3 Member Data Documentation

6.46.3.1 AllowMultipleTargets

```
bool AllowMultipleTargets
```

Is it allowed to select multiple targets for the button?

6.46.3.2 Label

```
string Label
```

Button label.

6.46.3.3 TargetMethod

```
string TargetMethod
```

The method to invoke when the button is clicked.

6.47 FieldsMask< T > Class Template Reference

Base class for [FieldsMask<T>](#).

Inherits FieldsMask.

Public Member Functions

- [FieldsMask](#) ()
Constructor for [FieldsMask<T>](#).
- [FieldsMask](#) (Func< [Mask256](#) > getDefaultsDelegate)
Constructor for [FieldsMask](#).
- [FieldsMask](#) (long maskA, long maskB=0, long maskC=0, long maskD=0)
Constructor for [FieldsMask](#).
- [FieldsMask](#) ([Mask256](#) mask)
Constructor for [FieldsMask<T>](#).

Static Public Member Functions

- static implicit [operator Mask256](#) ([FieldsMask](#) mask)
Implicitly convert [FieldsMask](#) to its long mask value.

Public Attributes

- [Mask256](#) Mask
The internal mask value.

Protected Member Functions

- [FieldsMask](#) ()
Constructor for [FieldsMask](#).
- [FieldsMask](#) (long a, long b, long c, long d)
Constructor for [FieldsMask](#).
- [FieldsMask](#) ([Mask256](#) mask)
Constructor for [FieldsMask](#).

6.47.1 Detailed Description

Base class for [FieldsMask<T>](#).

Associates and displays a 64 bit mask which represents the field members of a struct. Makes it possible to treat a Struct like an Flags Enum. NOTE: A [FieldsMask<T>](#) attribute is required for proper rendering in the Inspector.

6.47.2 Constructor & Destructor Documentation

6.47.2.1 FieldsMask() [1/7]

```
FieldsMask (
    Mask256 mask ) [protected]
```

Constructor for [FieldsMask](#).

6.47.2.2 FieldsMask() [2/7]

```
FieldsMask (
    long a,
    long b,
    long c,
    long d ) [protected]
```

Constructor for [FieldsMask](#).

6.47.2.3 FieldsMask() [3/7]

```
FieldsMask ( ) [protected]
```

Constructor for [FieldsMask](#).

6.47.2.4 FieldsMask() [4/7]

```
FieldsMask (
    Mask256 mask )
```

Constructor for [FieldsMask<T>](#).

6.47.2.5 FieldsMask() [5/7]

```
FieldsMask (
    long maskA,
    long maskB = 0,
    long maskC = 0,
    long maskD = 0 )
```

Constructor for [FieldsMask](#).

6.47.2.6 FieldsMask() [6/7]

```
FieldsMask ( )
```

Constructor for [FieldsMask<T>](#).

6.47.2.7 FieldsMask() [7/7]

```
FieldsMask (
    Func< Mask256 > getDefaultsDelegate )
```

Constructor for [FieldsMask](#).

6.47.3 Member Function Documentation

6.47.3.1 operator Mask256()

```
static implicit operator Mask256 (
    FieldsMask< T > mask ) [static]
```

Implicitly convert [FieldsMask](#) to its long mask value.

6.47.4 Member Data Documentation

6.47.4.1 Mask

[Mask256](#) Mask

The internal mask value.

6.48 FixedArray< T > Class Template Reference

A fixed size array that can be used in structs.

Inherits IEnumerable< T >.

Classes

- struct [Enumerator](#)
Enumerator for the [FixedArray](#) struct.

Public Member Functions

- void [Clear](#) ()
Sets all elements in the array to their default value.
- void [CopyFrom](#) (List< T > source, int sourceOffset, int sourceCount)
Copies a range of elements from a source list into the [FixedArray](#).
- void [CopyFrom](#) (T[] source, int sourceOffset, int sourceCount)
Copies a range of elements from a source array into the [FixedArray](#).
- void [CopyTo](#) (List< T > list)
Adds each value to the supplied List. This does not clear the list, so values will be appended to the existing list.
- void [CopyTo](#) (T[] array, bool throwIfOverflow=true)
Copies values to the supplied array.
- [FixedArray](#) (T *array, int length)
NetworkArray constructor.
- [Enumerator](#) [GetEnumerator](#) ()
Returns an enumerator that iterates through the [FixedArray](#).
- IEnumerable< T > IEnumerable< T >. [GetEnumerator](#) ()
- IEnumerable IEnumerable. [GetEnumerator](#) ()
- T[] [ToArray](#) ()
Allocates a new array and copies values from this array. For a non-alloc alternative use [CopyTo\(List< T >\)](#).
- string [ToListString](#) ()
Returns the elements of this array as a string, with value separated by characters. Specifically for use in the Unity inspector. This is private and only is found by NetworkBehaviourEditor using reflection, so do not rename this method.
- override string [ToString](#) ()
Returns a string that represents the current object.

Static Public Member Functions

- static unsafe [FixedArray](#)< T > [Create](#)< T > (ref T firstField, int length)
Creates a [FixedArray](#) with a specified length.
- static unsafe [FixedArray](#)< TAdapted > [Create](#)< TActual, TAdapted > (ref TActual firstField, int length)
Creates a [FixedArray](#) with a specified length and adapts the type of the elements.
- static unsafe [FixedArray](#)< T > [CreateFromFieldSequence](#)< T > (ref T firstField, ref T lastField)
Creates a [FixedArray](#) from a sequence of fields.
- static int [IndexOf](#)< T > (this [FixedArray](#)< T > array, T elem)
Returns the index of the first occurrence of a value in the [FixedArray](#).

Public Attributes

- T * [_array](#)
- int [_length](#)

Static Public Attributes

- static [StringBuilder](#) [_stringBuilderCached](#)

Properties

- int [Length](#) [get]
The fixed size of the array.
- ref T [this\[int index\]](#) [get]
Indexer of array elements.

6.48.1 Detailed Description

A fixed size array that can be used in structs.

Helper methods for [FixedArray](#).

Template Parameters

<i>T</i>	
----------	--

Type Constraints

T: **unmanaged**

6.48.2 Constructor & Destructor Documentation

6.48.2.1 FixedArray()

```
FixedArray (
    T * array,
    int length )
```

[NetworkArray](#) constructor.

6.48.3 Member Function Documentation

6.48.3.1 Clear()

```
void Clear ( )
```

Sets all elements in the array to their default value.

6.48.3.2 CopyFrom() [1/2]

```
void CopyFrom (
    List< T > source,
    int sourceOffset,
    int sourceCount )
```

Copies a range of elements from a source list into the [FixedArray](#).

Parameters

<i>source</i>	The source list from which to copy elements. Must not be null.
<i>sourceOffset</i>	The zero-based index in the source list at which copying begins.
<i>sourceCount</i>	The number of elements to copy from the source list.

Exceptions

<i>ArgumentNullException</i>	Thrown when the provided source list is null.
<i>ArgumentException</i>	Thrown when the number of elements to copy is greater than the length of the FixedArray .
<i>ArgumentOutOfRangeException</i>	Thrown when the sum of sourceOffset and sourceCount is greater than the length of the source list.

6.48.3.3 CopyFrom() [2/2]

```
void CopyFrom (
```

```
T[] source,
int sourceOffset,
int sourceCount )
```

Copies a range of elements from a source array into the [FixedArray](#).

Parameters

<i>source</i>	The source array from which to copy elements. Must not be null.
<i>sourceOffset</i>	The zero-based index in the source array at which copying begins.
<i>sourceCount</i>	The number of elements to copy from the source array.

Exceptions

<i>ArgumentNullException</i>	Thrown when the provided source array is null.
<i>ArgumentException</i>	Thrown when the number of elements to copy is greater than the length of the FixedArray .
<i>ArgumentOutOfRangeException</i>	Thrown when the sum of sourceOffset and sourceCount is greater than the length of the source array.

6.48.3.4 CopyTo() [1/2]

```
void CopyTo (
    List< T > list )
```

Adds each value to the supplied List. This does not clear the list, so values will be appended to the existing list.

6.48.3.5 CopyTo() [2/2]

```
void CopyTo (
    T[] array,
    bool throwIfOverflow = true )
```

Copies values to the supplied array.

Parameters

<i>array</i>	The array to copy values to. Must not be null.
<i>throwIfOverflow</i>	If true, this method will throw an error if the supplied array is smaller than this NetworkArray<T> . If false, will only copy as many elements as the target array can hold.

6.48.3.6 Create< T >()

```
static unsafe FixedArray<T> Create< T > (  
    ref T firstField,  
    int length ) [static]
```

Creates a [FixedArray](#) with a specified length.

Parameters

<i>firstField</i>	Reference to the first field in the array.
<i>length</i>	The length of the array.

Returns

A new [FixedArray](#) instance with the specified length.

Type Constraints

T* : *unmanaged

6.48.3.7 Create< TActual, TAdapted >()

```
static unsafe FixedArray<TAdapted> Create< TActual, TAdapted > (  
    ref TActual firstField,  
    int length ) [static]
```

Creates a [FixedArray](#) with a specified length and adapts the type of the elements.

Parameters

<i>firstField</i>	Reference to the first field in the array.
<i>length</i>	The length of the array.

Returns

A new [FixedArray](#) instance with the specified length and adapted type of elements.

Type Constraints

TActual* : *unmanaged

TAdapted* : *unmanaged

6.48.3.8 CreateFromFieldSequence< T >()

```
static unsafe FixedArray<T> CreateFromFieldSequence< T > (
    ref T firstField,
    ref T lastField ) [static]
```

Creates a [FixedArray](#) from a sequence of fields.

Parameters

<i>firstField</i>	Reference to the first field in the sequence.
<i>lastField</i>	Reference to the last field in the sequence.

Returns

A new [FixedArray](#) instance with the values from the sequence of fields.

Type Constraints

***T*: unmanaged**

6.48.3.9 GetEnumerator()

```
Enumerator GetEnumerator ( )
```

Returns an enumerator that iterates through the [FixedArray](#).

6.48.3.10 IndexOf< T >()

```
static int IndexOf< T > (
    this FixedArray< T > array,
    T elem ) [static]
```

Returns the index of the first occurrence of a value in the [FixedArray](#).

Parameters

<i>array</i>	The FixedArray to search.
<i>elem</i>	The value to locate in the FixedArray .

Returns

The zero-based index of the first occurrence of *elem* within the entire [FixedArray](#), if found; otherwise, -1.

Type Constraints

***T*: unmanaged**

T: *IEquatable*<T>

6.48.3.11 ToArray()

```
T [] ToArray ( )
```

Allocates a new array and copies values from this array. For a non-alloc alternative use [CopyTo\(List<T>\)](#).

6.48.3.12 ToListString()

```
string ToListString ( )
```

Returns the elements of this array as a string, with value separated by characters. Specifically for use in the Unity inspector. This is private and only is found by NetworkBehaviourEditor using reflection, so do not rename this method.

6.48.3.13 ToString()

```
override string ToString ( )
```

Returns a string that represents the current object.

6.48.4 Property Documentation

6.48.4.1 Length

```
int Length [get]
```

The fixed size of the array.

6.48.4.2 this[int index]

```
ref T this[int index] [get]
```

Indexer of array elements.

6.49 FixedArray< T >.Enumerator Struct Reference

[Enumerator](#) for the [FixedArray](#) struct.

Inherits [IEnumerator< T >](#).

Public Member Functions

- void [Dispose](#) ()
Performs application-defined tasks associated with freeing, releasing, or resetting unmanaged resources.
- [Enumerator](#) ([FixedArray< T > array](#))
Initializes a new instance of the [Enumerator](#) struct.
- bool [MoveNext](#) ()
Advances the enumerator to the next element of the collection.
- void [Reset](#) ()
Sets the enumerator to its initial position, which is before the first element in the collection.

Public Attributes

- [FixedArray< T > _array](#)
- int [_index](#)

Properties

- T [Current](#) [get]
Gets the current element in the collection.
- object [IEnumerator](#). [Current](#) [get]

6.49.1 Detailed Description

[Enumerator](#) for the [FixedArray](#) struct.

6.49.2 Constructor & Destructor Documentation

6.49.2.1 Enumerator()

```
Enumerator (  
    FixedArray< T > array )
```

Initializes a new instance of the [Enumerator](#) struct.

Parameters

<code>array</code>	The FixedArray instance to enumerate.
--------------------	---

6.49.3 Member Function Documentation

6.49.3.1 Dispose()

```
void Dispose ( )
```

Performs application-defined tasks associated with freeing, releasing, or resetting unmanaged resources.

6.49.3.2 MoveNext()

```
bool MoveNext ( )
```

Advances the enumerator to the next element of the collection.

Returns

True if the enumerator was successfully advanced to the next element; false if the enumerator has passed the end of the collection.

6.49.3.3 Reset()

```
void Reset ( )
```

Sets the enumerator to its initial position, which is before the first element in the collection.

6.49.4 Property Documentation

6.49.4.1 Current

```
T Current [get]
```

Gets the current element in the collection.

6.50 FixedBufferPropertyAttribute Class Reference

Fixed Buffer Property Attribute

Inherits [PropertyAttribute](#).

Public Member Functions

- [FixedBufferPropertyAttribute](#) ([Type](#) fieldType, [Type](#) surrogateType, int capacity)
FixedBufferPropertyAttribute Constructor

Properties

- int [Capacity](#) [get]
Fixed Buffer Capacity
- [Type](#) [SurrogateType](#) [get]
Fixed Buffer Surrogate Type
- [Type](#) [Type](#) [get]
Fixed Buffer Type

6.50.1 Detailed Description

Fixed Buffer Property Attribute

6.50.2 Constructor & Destructor Documentation

6.50.2.1 FixedBufferPropertyAttribute()

```
FixedBufferPropertyAttribute (
    Type fieldType,
    Type surrogateType,
    int capacity )
```

[FixedBufferPropertyAttribute](#) Constructor

Parameters

<i>fieldType</i>	Type
<i>surrogateType</i>	SurrogateType
<i>capacity</i>	Capacity

6.50.3 Property Documentation

6.50.3.1 Capacity

`int Capacity [get]`

Fixed Buffer Capacity

6.50.3.2 SurrogateType

`Type SurrogateType [get]`

Fixed Buffer Surrogate Type

6.50.3.3 Type

`Type Type [get]`

Fixed Buffer Type

6.51 FixedStorage Class Reference

Provides utility methods for fixed storage types.

Static Public Member Functions

- static int [GetWordCount< T > \(\)](#)
Gets the word count of a fixed storage type.

6.51.1 Detailed Description

Provides utility methods for fixed storage types.

6.51.2 Member Function Documentation

6.51.2.1 GetWordCount< T >()

`static int GetWordCount< T > () [static]`

Gets the word count of a fixed storage type.

Template Parameters

<i>T</i>	The type of the fixed storage.
----------	--------------------------------

Returns

The word count of the fixed storage type.

Type Constraints

T: *unmanaged*

T: *IFixedStorage*

6.52 FloatCompressed Struct Reference

Represents a compressed float value for network transmission.

Inherits [INetworkStruct](#), and [IEquatable< FloatCompressed >](#).

Public Member Functions

- bool [Equals](#) ([FloatCompressed](#) other)
Checks if the current [FloatCompressed](#) instance is equal to the other [FloatCompressed](#) instance.
- override bool [Equals](#) (object obj)
Checks if the provided object is a [FloatCompressed](#) instance and if it's equal to the current [FloatCompressed](#) instance.
- override int [GetHashCode](#) ()
Returns the hash code for the current [FloatCompressed](#) instance.

Static Public Member Functions

- static implicit [operator float](#) ([FloatCompressed](#) q)
Implicit conversion from [FloatCompressed](#) to float.
- static implicit [operator FloatCompressed](#) (float v)
Implicit conversion from float to [FloatCompressed](#).
- static bool [operator!=](#) ([FloatCompressed](#) left, [FloatCompressed](#) right)
Inequality operator for [FloatCompressed](#) struct.
- static bool [operator==](#) ([FloatCompressed](#) left, [FloatCompressed](#) right)
Equality operator for [FloatCompressed](#) struct.

Public Attributes

- int [valueEncoded](#)
Encoded value of the float.

6.52.1 Detailed Description

Represents a compressed float value for network transmission.

6.52.2 Member Function Documentation

6.52.2.1 Equals() [1/2]

```
bool Equals (  
    FloatCompressed other )
```

Checks if the current [FloatCompressed](#) instance is equal to the other [FloatCompressed](#) instance.

Parameters

<i>other</i>	The other FloatCompressed instance to compare with the current FloatCompressed instance.
--------------	--

Returns

True if the values of both [FloatCompressed](#) instances are equal, otherwise false.

6.52.2.2 Equals() [2/2]

```
override bool Equals (  
    object obj )
```

Checks if the provided object is a [FloatCompressed](#) instance and if it's equal to the current [FloatCompressed](#) instance.

Parameters

<i>obj</i>	The object to compare with the current FloatCompressed instance.
------------	--

Returns

True if the provided object is a [FloatCompressed](#) instance and it's equal to the current [FloatCompressed](#) instance, otherwise false.

6.52.2.3 GetHashCode()

```
override int GetHashCode ( )
```

Returns the hash code for the current [FloatCompressed](#) instance.

Returns

A hash code for the current [FloatCompressed](#) instance.

6.52.2.4 operator float()

```
static implicit operator float (
    FloatCompressed q ) [static]
```

Implicit conversion from [FloatCompressed](#) to float.

Parameters

<i>q</i>	The FloatCompressed instance to convert.
----------	--

Returns

The decompressed float value of the [FloatCompressed](#) instance.

6.52.2.5 operator FloatCompressed()

```
static implicit operator FloatCompressed (
    float v ) [static]
```

Implicit conversion from float to [FloatCompressed](#).

Parameters

<i>v</i>	The float value to convert.
----------	-----------------------------

Returns

A new [FloatCompressed](#) instance with the compressed value of the float.

6.52.2.6 operator"!=()

```
static bool operator!= (
    FloatCompressed left,
    FloatCompressed right ) [static]
```

Inequality operator for [FloatCompressed](#) struct.

Parameters

<i>left</i>	First FloatCompressed instance.
<i>right</i>	Second FloatCompressed instance.

Returns

True if the value of the first [FloatCompressed](#) instance is not equal to the value of the second [FloatCompressed](#) instance, otherwise false.

6.52.2.7 operator==()

```
static bool operator== (
    FloatCompressed left,
    FloatCompressed right ) [static]
```

Equality operator for [FloatCompressed](#) struct.

Parameters

<i>left</i>	First FloatCompressed instance.
<i>right</i>	Second FloatCompressed instance.

Returns

True if the value of the first [FloatCompressed](#) instance is equal to the value of the second [FloatCompressed](#) instance, otherwise false.

6.52.3 Member Data Documentation**6.52.3.1 valueEncoded**

```
int valueEncoded
```

Encoded value of the float.

6.53 FloatUtils Class Reference

Provides utility methods for compressing and decompressing float values.

Static Public Member Functions

- static unsafe int [Compress](#) (float f, int accuracy=[DEFAULT_ACCURACY](#))
Compresses a float value.
- static unsafe float [Decompress](#) (int value, float accuracy=[DEFAULT_ACCURACY](#))
Decompresses a compressed float value.

Static Public Attributes

- const int `DEFAULT_ACCURACY` = 1 << 10
Default accuracy for float compression and decompression.

6.53.1 Detailed Description

Provides utility methods for compressing and decompressing float values.

6.53.2 Member Function Documentation

6.53.2.1 Compress()

```
static unsafe int Compress (
    float f,
    int accuracy = DEFAULT_ACCURACY ) [static]
```

Compresses a float value.

Parameters

<i>f</i>	The float value to compress.
<i>accuracy</i>	The accuracy to use for compression. Defaults to <code>DEFAULT_ACCURACY</code> .

Returns

The compressed float value.

6.53.2.2 Decompress()

```
static unsafe float Decompress (
    int value,
    float accuracy = DEFAULT_ACCURACY ) [static]
```

Decompresses a compressed float value.

Parameters

<i>value</i>	The compressed float value to decompress.
<i>accuracy</i>	The accuracy to use for decompression. Defaults to <code>DEFAULT_ACCURACY</code> .

Returns

The decompressed float value.

6.53.3 Member Data Documentation**6.53.3.1 DEFAULT_ACCURACY**

```
const int DEFAULT_ACCURACY = 1 << 10 [static]
```

Default accuracy for float compression and decompression.

6.54 FusionGlobalScriptableObject< T > Class Template Reference

A base class for ScriptableObjects that are meant to be globally accessible, at edit-time and runtime. The way such objects are loaded is driven by usages of [FusionGlobalScriptableObjectSourceAttribute](#) attributes.

Inherits [FusionScriptableObject](#), and [FusionGlobalScriptableObject](#).

Protected Member Functions

- virtual void [OnDisable](#) ()
 - If the current instance is global, unsets [IsGlobal](#) and calls [OnUnloadedAsGlobal](#)*
- virtual void [OnLoadedAsGlobal](#) ()
 - Invoked when the instance is loaded as global.*
- virtual void [OnUnloadedAsGlobal](#) (bool destroyed)
 - Invoked when the instance is unloaded as global.*

Static Protected Member Functions

- static bool [TryGetGlobalInternal](#) (out T global)
 - Loads or returns the current global instance. Returns null if loading an instance failed.*
- static bool [UnloadGlobalInternal](#) ()
 - Unloads the global instance if it is loaded.*

Properties

- static T [GlobalInternal](#) [get, set]
 - A singleton instance-like property. Loads or returns the current global instance. Derived classes can package it in a property with a different name. Throws if loading an instance failed.*
- bool [IsGlobal](#) [get]
 - Is this instance a global instance.*
- static bool [IsGlobalLoadedInternal](#) [get]
 - Returns true if a global instance is loaded. Compared to [GlobalInternal](#), it does not attempt to load an instance.*

6.54.1 Detailed Description

A base class for ScriptableObject that are meant to be globally accessible, at edit-time and runtime. The way such objects are loaded is driven by usages of [FusionGlobalScriptableObjectSourceAttribute](#) attributes.

Type Constraints

T: [FusionGlobalScriptableObject](#)<*T*>

6.54.2 Member Function Documentation

6.54.2.1 OnDisable()

```
virtual void OnDisable ( ) [protected], [virtual]
```

If the current instance is global, unsets [IsGlobal](#) and calls [OnUnloadedAsGlobal](#)

Reimplemented in [NetworkProjectConfigAsset](#).

6.54.2.2 OnLoadedAsGlobal()

```
virtual void OnLoadedAsGlobal ( ) [protected], [virtual]
```

Invoked when the instance is loaded as global.

6.54.2.3 OnUnloadedAsGlobal()

```
virtual void OnUnloadedAsGlobal (
    bool destroyed ) [protected], [virtual]
```

Invoked when the instance is unloaded as global.

Parameters

<i>destroyed</i>	
------------------	--

6.54.2.4 TryGetGlobalInternal()

```
static bool TryGetGlobalInternal (
    out T global ) [static], [protected]
```

Loads or returns the current global instance. Returns `null` if loading an instance failed.

Parameters

<i>global</i>	
---------------	--

Returns

6.54.2.5 UnloadGlobalInternal()

```
static bool UnloadGlobalInternal ( ) [static], [protected]
```

Unloads the global instance if it is loaded.

Returns

`true` if an instance was unloaded

6.54.3 Property Documentation

6.54.3.1 GlobalInternal

```
T GlobalInternal [static], [get], [set], [protected]
```

A singleton instance-like property. Loads or returns the current global instance. Derived classes can package it in a property with a different name. Throws if loading an instance failed.

Exceptions

<i>InvalidOperationException</i>	
----------------------------------	--

6.54.3.2 IsGlobal

```
bool IsGlobal [get]
```

Is this instance a global instance.

6.54.3.3 IsGlobalLoadedInternal

```
bool IsGlobalLoadedInternal [static], [get], [protected]
```

Returns true if a global instance is loaded. Compared to [GlobalInternal](#), it does not attempt to load an instance.

6.55 FusionGlobalScriptableObjectAttribute Class Reference

Provides additional information for a global scriptable object.

Inherits [Attribute](#).

Public Member Functions

- [FusionGlobalScriptableObjectAttribute](#) (string defaultPath)
Creates a new instance.

Properties

- string [DefaultContents](#) [get, set]
The default contents for the asset, if it is a TextAsset.
- string [DefaultContentsGeneratorMethod](#) [get, set]
Name of the method that is used to generate the default contents for the asset.
- string [DefaultPath](#) [get]
The default path for the asset.

6.55.1 Detailed Description

Provides additional information for a global scriptable object.

6.55.2 Constructor & Destructor Documentation

6.55.2.1 FusionGlobalScriptableObjectAttribute()

```
FusionGlobalScriptableObjectAttribute (
    string defaultPath )
```

Creates a new instance.

Parameters

<i>defaultPath</i>	The default path for the asset.
--------------------	---------------------------------

6.55.3 Property Documentation

6.55.3.1 DefaultContents

```
string DefaultContents [get], [set]
```

The default contents for the asset, if it is a TextAsset.

6.55.3.2 DefaultContentsGeneratorMethod

```
string DefaultContentsGeneratorMethod [get], [set]
```

Name of the method that is used to generate the default contents for the asset.

6.55.3.3 DefaultPath

```
string DefaultPath [get]
```

The default path for the asset.

6.56 FusionGlobalScriptableObjectLoadResult Struct Reference

The result of [FusionGlobalScriptableObjectSourceAttribute.Load](#). Contains the loaded object and an optional unloader delegate.

Public Member Functions

- [FusionGlobalScriptableObjectLoadResult](#) ([FusionGlobalScriptableObject](#) obj, [FusionGlobalScriptableObjectUnloadDelegate](#) unloader=null)

Static Public Member Functions

- static implicit [operator FusionGlobalScriptableObjectLoadResult](#) ([FusionGlobalScriptableObject](#) result)
Implicitly converts a [FusionGlobalScriptableObject](#) to a [FusionGlobalScriptableObjectLoadResult](#).

Public Attributes

- readonly [FusionGlobalScriptableObject](#) [Object](#)
Object instance.
- readonly [FusionGlobalScriptableObjectUnloadDelegate](#) [Unloader](#)
An optional delegate that is used to unload [Object](#).

6.56.1 Detailed Description

The result of [FusionGlobalScriptableObjectSourceAttribute.Load](#). Contains the loaded object and an optional unloader delegate.

6.56.2 Constructor & Destructor Documentation

6.56.2.1 FusionGlobalScriptableObjectLoadResult()

```
FusionGlobalScriptableObjectLoadResult (
    FusionGlobalScriptableObject obj,
    FusionGlobalScriptableObjectUnloadDelegate unloader = null )
```

Parameters

<i>obj</i>	Object instance.
<i>unloader</i>	An optional delegate that is used to unload <i>obj</i> .

6.56.3 Member Function Documentation

6.56.3.1 operator FusionGlobalScriptableObjectLoadResult()

```
static implicit operator FusionGlobalScriptableObjectLoadResult (
    FusionGlobalScriptableObject result ) [static]
```

Implicitly converts a [FusionGlobalScriptableObject](#) to a [FusionGlobalScriptableObjectLoadResult](#).

6.56.4 Member Data Documentation

6.56.4.1 Object

```
readonly FusionGlobalScriptableObject Object
```

Object instance.

6.56.4.2 Unloader

readonly FusionGlobalScriptableObjectUnloadDelegate Unloader

An optional delegate that is used to unload [Object](#).

6.57 FusionGlobalScriptableObjectSourceAttribute Class Reference

Base class for all attributes that can be used to load [FusionGlobalScriptableObject](#). Attributes need to be registered at the assembly level. For instance, this snippet is used to register a default loader, that attempts to load from Resources based on [FusionGlobalScriptableObjectAttribute.DefaultPath](#):

Inherits [Attribute](#).

Public Member Functions

- [FusionGlobalScriptableObjectSourceAttribute](#) (Type objectType)

Parameters

objectType	Type or the base type of FusionGlobalScriptableObject that this loader supports.
------------	--

- abstract [FusionGlobalScriptableObjectLoadResult Load](#) (Type type)
Attempt to load the object of the specified type. Return `default` if the object cannot be loaded.

Properties

- bool [AllowEditMode](#) = false [get, set]
Can this loader be used in edit mode.
- bool [AllowFallback](#) = false [get, set]
Does this loader allow fallback to the next loader?
- Type [ObjectType](#) [get]
Type or the base type of [FusionGlobalScriptableObject](#) that this loader supports.
- int [Order](#) [get, set]
Order in which this loader will be executed. Lower values are executed first.

6.57.1 Detailed Description

Base class for all attributes that can be used to load [FusionGlobalScriptableObject](#). Attributes need to be registered at the assembly level. For instance, this snippet is used to register a default loader, that attempts to load from Resources based on [FusionGlobalScriptableObjectAttribute.DefaultPath](#):

```
[assembly: Fusion.FusionGlobalScriptableObjectResource (typeof (Fusion.FusionGlobalScriptableObjectSourceAttribute),
Order = 2000, AllowFallback = true)]
```

6.57.2 Member Function Documentation

6.57.2.1 Load()

```
abstract FusionGlobalScriptableObjectLoadResult Load (  
    Type type ) [pure virtual]
```

Attempt to load the object of the specified type. Return `default` if the object cannot be loaded.

Parameters

<i>type</i>	The requested type
-------------	--------------------

6.57.3 Property Documentation

6.57.3.1 AllowEditMode

```
bool AllowEditMode = false [get], [set]
```

Can this loader be used in edit mode.

6.57.3.2 AllowFallback

```
bool AllowFallback = false [get], [set]
```

Does this loader allow fallback to the next loader?

6.57.3.3 ObjectType

```
Type ObjectType [get]
```

Type or the base type of [FusionGlobalScriptableObject](#) that this loader supports.

6.57.3.4 Order

```
int Order [get], [set]
```

Order in which this loader will be executed. Lower values are executed first.

6.58 FusionMonoBehaviour Class Reference

Base class for all [Fusion](#) MonoBehaviours.

Inherits MonoBehaviour.

6.58.1 Detailed Description

Base class for all [Fusion](#) MonoBehaviours.

6.59 FusionScriptableObject Class Reference

Base class for all [Fusion](#) scriptable objects.

Inherits ScriptableObject.

Inherited by [FusionGlobalScriptableObject< NetworkProjectConfigAsset >](#), and [FusionGlobalScriptableObject< T >](#).

6.59.1 Detailed Description

Base class for all [Fusion](#) scriptable objects.

6.60 HeapConfiguration Class Reference

Memory Heap Settings

Inherits IConfigurationSanityCheck.

Public Member Functions

- [HeapConfiguration Init](#) (int globalsSize)
Initializes and creates a new [HeapConfiguration](#) based on the Global Size
- void [SanityCheck](#) ()
- override string [ToString](#) ()
ToString

Public Attributes

- int [GlobalsSize](#)
Heap Global Size
- int [PageCount](#) = [Allocator.Config.DEFAULT_BLOCK_COUNT](#)
Default number of Heap Pages
- [PageSizes PageShift](#) = [Allocator.Config.DEFAULT_BLOCK_SHIFT](#)
Default size of each Heap Page

6.60.1 Detailed Description

Memory Heap Settings

6.60.2 Member Function Documentation

6.60.2.1 Init()

```
HeapConfiguration Init (  
    int globalsSize )
```

Initializes and creates a new [HeapConfiguration](#) based on the Global Size

6.60.2.2 ToString()

```
override string ToString ( )
```

ToString

6.60.3 Member Data Documentation

6.60.3.1 GlobalsSize

```
int GlobalsSize
```

Heap Global Size

6.60.3.2 PageCount

```
int PageCount = Allocator.Config.DEFAULT\_BLOCK\_COUNT
```

Default number of Heap Pages

6.60.3.3 PageShift

```
PageSizes PageShift = Allocator.Config.DEFAULT_BLOCK_SHIFT
```

Default size of each Heap Page

6.61 HideArrayElementLabelAttribute Class Reference

Attribute used to hide the label of an array element in the inspector.

Inherits [DecoratingPropertyAttribute](#).

Public Member Functions

- [HideArrayElementLabelAttribute](#) ()

Initializes a new instance of the [HideArrayElementLabelAttribute](#) class.

Additional Inherited Members

6.61.1 Detailed Description

Attribute used to hide the label of an array element in the inspector.

6.61.2 Constructor & Destructor Documentation

6.61.2.1 HideArrayElementLabelAttribute()

```
HideArrayElementLabelAttribute ( )
```

Initializes a new instance of the [HideArrayElementLabelAttribute](#) class.

6.62 Hitbox Class Reference

Represents a single lag-compensated collider. Multiple component instances can be added anywhere in the hierarchy of a [NetworkObject](#) which includes a [HitboxRoot](#).

Inherits [Behaviour](#).

Public Member Functions

- void [OnDrawGizmos](#) ()
Draws this hitbox gizmo on Unity editor.
- void [SetLayer](#) (int layer)
Set a layer mask for this [Hitbox](#) gameobject. This method caches the layer mask.

Public Attributes

- Vector3 [BoxExtents](#)
When [Type](#) is set to [HitboxTypes.Box](#), this defines the local-space geometry for narrow-phase checks.
- float [CapsuleExtents](#)
When [Type](#) is set to [HitboxTypes.Capsule](#), this defines the local-space geometry for narrow-phase checks.
- float [CapsuleRadius](#)
When [Type](#) is set to [HitboxTypes.Capsule](#), this defines the local-space geometry for narrow-phase checks.
- Color [GizmosColor](#) = Color.yellow
Color used when drawing gizmos for this hitbox.
- Vector3 [Offset](#)
This [Hitbox](#)'s local-space offset from its [GameObject](#) position.
- [HitboxRoot](#) [Root](#)
Reference to the top-level [HitboxRoot](#) component for this [NetworkObject](#).
- float [SphereRadius](#)
When [Type](#) is set to [HitboxTypes.Sphere](#), this defines the local-space geometry for narrow-phase checks.
- [HitboxTypes](#) [Type](#)
The collision geometry type for this [Hitbox](#).

Protected Member Functions

- virtual void [DrawGizmos](#) (Color color, ref Matrix4x4 localToWorldMatrix)
Draw the [Hitbox](#) gizmos.

Properties

- int [ColliderIndex](#) [get]
Index assigned to the collider of this hitbox on the lag-compensated snapshots.
- bool [HitboxActive](#) [get, set]
Get or set the state of this [Hitbox](#). If a hitbox or its [HitboxRoot](#) are not active, it will not be hit by lag-compensated queries.
- Int32 [HitboxIndex](#) [get]
The index of this hitbox in the [HitboxRoot.Hitboxes](#) array on [Root](#). The value is set by the root when initializing the nested hitboxes with [HitboxRoot.InitHitboxes](#).
- Vector3 [Position](#) [get]
World-space position (includes [Offset](#)) of this [Hitbox](#).

Additional Inherited Members

6.62.1 Detailed Description

Represents a single lag-compensated collider. Multiple component instances can be added anywhere in the hierarchy of a [NetworkObject](#) which includes a [HitboxRoot](#).

6.62.2 Member Function Documentation

6.62.2.1 DrawGizmos()

```
virtual void DrawGizmos (
    Color color,
    ref Matrix4x4 localToWorldMatrix ) [protected], [virtual]
```

Draw the [Hitbox](#) gizmos.

6.62.2.2 OnDrawGizmos()

```
void OnDrawGizmos ( )
```

Draws this hitbox gizmo on Unity editor.

6.62.2.3 SetLayer()

```
void SetLayer (
    int layer )
```

Set a layer mask for this [Hitbox](#) gameobject. This method caches the layer mask.

Parameters

<i>layer</i>	
--------------	--

6.62.3 Member Data Documentation

6.62.3.1 BoxExtents

```
Vector3 BoxExtents
```

When [Type](#) is set to [HitboxTypes.Box](#), this defines the local-space geometry for narrow-phase checks.

6.62.3.2 CapsuleExtents

```
float CapsuleExtents
```

When [Type](#) is set to [HitboxTypes.Capsule](#), this defines the local-space geometry for narrow-phase checks.

6.62.3.3 CapsuleRadius

```
float CapsuleRadius
```

When [Type](#) is set to [HitboxTypes.Capsule](#), this defines the local-space geometry for narrow-phase checks.

6.62.3.4 GizmosColor

```
Color GizmosColor = Color.yellow
```

Color used when drawing gizmos for this hitbox.

6.62.3.5 Offset

```
Vector3 Offset
```

This [Hitbox](#)'s local-space offset from its [GameObject](#) position.

6.62.3.6 Root

```
HitboxRoot Root
```

Reference to the top-level [HitboxRoot](#) component for this [NetworkObject](#).

6.62.3.7 SphereRadius

```
float SphereRadius
```

When [Type](#) is set to [HitboxTypes.Sphere](#), this defines the local-space geometry for narrow-phase checks.

6.62.3.8 Type

`HitboxTypes` Type

The collision geometry type for this [Hitbox](#).

6.62.4 Property Documentation

6.62.4.1 ColliderIndex

```
int ColliderIndex [get]
```

Index assigned to the collider of this hitbox on the lag-compensated snapshots.

6.62.4.2 HitboxActive

```
bool HitboxActive [get], [set]
```

Get or set the state of this [Hitbox](#). If a hitbox or its [HitboxRoot](#) are not active, it will not be hit by lag-compensated queries.

6.62.4.3 HitboxIndex

```
Int32 HitboxIndex [get]
```

The index of this hitbox in the [HitboxRoot.Hitboxes](#) array on [Root](#). The value is set by the root when initializing the nested hitboxes with [HitboxRoot.InitHitboxes](#).

6.62.4.4 Position

```
Vector3 Position [get]
```

World-space position (includes Offset) of this [Hitbox](#).

6.63 HitboxManager Class Reference

Entry point for lag compensated [Hitbox](#) queries, which maintains a history buffer, and provides lag compensated raycast and overlap methods. Singleton instance is accessible through the property `Runner.LagCompensation`.

Inherits [SimulationBehaviour](#), [IAfterTick](#), [IBeforeSimulation](#), and [ISpawned](#).

Public Member Functions

- void [GetPlayerTickAndAlpha](#) ([PlayerRef](#) player, out int? tickFrom, out int? tickTo, out float? alpha)

Gets the tick and alpha interpolate values for a player.
- [LagCompensationStatisticsSnapshot](#) [GetStatisticsSnapshot](#) ()

Gets a snapshot of the lag compensation statistics.
- int [OverlapBox](#) ([BoxOverlapQuery](#) query, List< [LagCompensatedHit](#) > hits, bool clearHits=true)

Performs a lag-compensated box overlap query against all registered hitboxes. If the [HitOptions.IncludePhysX](#) or [HitOptions.IncludeBox2D](#) flag is indicated, query will also include static colliders, Unity colliders are recommended for static geometry, rather than Hitboxes.
- int [OverlapBox](#) (Vector3 center, Vector3 extents, Quaternion orientation, int tick, int? tickTo, float? alpha, List< [LagCompensatedHit](#) > hits, int layerMask=-1, [HitOptions](#) options=[HitOptions.None](#), bool clearHits=true, QueryTriggerInteraction queryTriggerInteraction=QueryTriggerInteraction.UseGlobal, PreProcessingDelegate preProcessRoots=null)

Performs a lag-compensated box overlap query against all registered hitboxes. If the [HitOptions.IncludePhysX](#) or [HitOptions.IncludeBox2D](#) flag is indicated, query will also include static colliders, Unity colliders are recommended for static geometry, rather than Hitboxes.
- int [OverlapBox](#) (Vector3 center, Vector3 extents, Quaternion orientation, [PlayerRef](#) player, List< [LagCompensatedHit](#) > hits, int layerMask=-1, [HitOptions](#) options=[HitOptions.None](#), bool clearHits=true, QueryTriggerInteraction queryTriggerInteraction=QueryTriggerInteraction.UseGlobal, PreProcessingDelegate preProcessRoots=null)

Performs a lag-compensated box overlap query against all registered hitboxes. If the [HitOptions.IncludePhysX](#) or [HitOptions.IncludeBox2D](#) flag is indicated, query will also include static colliders, Unity colliders are recommended for static geometry, rather than Hitboxes.
- int [OverlapSphere](#) ([SphereOverlapQuery](#) query, List< [LagCompensatedHit](#) > hits, bool clearHits=true)

Performs a lag-compensated sphere overlap query against all registered hitboxes. If the [HitOptions.IncludePhysX](#) or [HitOptions.IncludeBox2D](#) flag is indicated, query will also include static colliders, Unity colliders are recommended for static geometry, rather than Hitboxes.
- int [OverlapSphere](#) (Vector3 origin, float radius, int tick, int? tickTo, float? alpha, List< [LagCompensatedHit](#) > hits, int layerMask=-1, [HitOptions](#) options=[HitOptions.None](#), bool clearHits=true, QueryTriggerInteraction queryTriggerInteraction=QueryTriggerInteraction.UseGlobal, PreProcessingDelegate preProcessRoots=null)

Performs a lag-compensated overlap sphere query against all registered hitboxes. If the [HitOptions.IncludePhysX](#) or [HitOptions.IncludeBox2D](#) flag is indicated, query will also include static colliders, Unity colliders are recommended for static geometry, rather than Hitboxes.
- int [OverlapSphere](#) (Vector3 origin, float radius, [PlayerRef](#) player, List< [LagCompensatedHit](#) > hits, int layerMask=-1, [HitOptions](#) options=[HitOptions.None](#), bool clearHits=true, QueryTriggerInteraction queryTriggerInteraction=QueryTriggerInteraction.UseGlobal, PreProcessingDelegate preProcessRoots=null)

Performs a lag-compensated overlap sphere query against all registered hitboxes. If the [HitOptions.IncludePhysX](#) or [HitOptions.IncludeBox2D](#) flag is indicated, query will also include static colliders, Unity colliders are recommended for static geometry, rather than Hitboxes.
- void [PositionRotation](#) ([Hitbox](#) hitbox, int tick, out Vector3 position, out Quaternion rotation, bool subTickAccuracy=false, int? tickTo=null, float? alpha=null)

Performs a lag-compensated query for a specific [Hitbox](#) position and rotation.
- void [PositionRotation](#) ([Hitbox](#) hitbox, [PlayerRef](#) player, out Vector3 position, out Quaternion rotation, bool subTickAccuracy=false)

Performs a lag-compensated query for a specific [Hitbox](#) position and rotation.
- bool [Raycast](#) ([RaycastQuery](#) query, out [LagCompensatedHit](#) hit)

Performs a lag-compensated raycast query against all registered hitboxes. If the [HitOptions.IncludePhysX](#) or [HitOptions.IncludeBox2D](#) flag is indicated, query will also include static colliders, Unity colliders are recommended for static geometry, rather than Hitboxes.
- bool [Raycast](#) (Vector3 origin, Vector3 direction, float length, int tick, int? tickTo, float? alpha, out [LagCompensatedHit](#) hit, int layerMask=-1, [HitOptions](#) options=[HitOptions.None](#), QueryTriggerInteraction queryTriggerInteraction=QueryTriggerInteraction.UseGlobal, PreProcessingDelegate preProcessRoots=null)

Performs a lag-compensated raycast query against all registered hitboxes. If the [HitOptions.IncludePhysX](#) or [HitOptions.IncludeBox2D](#) flag is indicated, query will also include static colliders, Unity colliders are recommended for static geometry, rather than Hitboxes.

- bool [Raycast](#) (Vector3 origin, Vector3 direction, float length, [PlayerRef](#) player, out [LagCompensatedHit](#) hit, int layerMask=-1, [HitOptions](#) options=[HitOptions.None](#), QueryTriggerInteraction queryTrigger↔ Interaction=QueryTriggerInteraction.UseGlobal, PreProcessingDelegate preProcessRoots=null)

Performs a lag-compensated raycast query against all registered hitboxes. If the [HitOptions.IncludePhysX](#) or [HitOptions.IncludeBox2D](#) flag is indicated, query will also include static colliders, Unity colliders are recommended for static geometry, rather than Hitboxes.
- int [RaycastAll](#) ([RaycastAllQuery](#) query, List< [LagCompensatedHit](#) > hits, bool clearHits=true)

Performs a lag-compensated raycast query against all registered hitboxes. If the [HitOptions.IncludePhysX](#) or [HitOptions.IncludeBox2D](#) flag is indicated, query will also include static colliders, Unity colliders are recommended for static geometry, rather than Hitboxes.
- int [RaycastAll](#) (Vector3 origin, Vector3 direction, float length, int tick, int? tickTo, float? alpha, List< [LagCompensatedHit](#) > hits, int layerMask=-1, bool clearHits=true, [HitOptions](#) options=[HitOptions.None](#), QueryTriggerInteraction queryTrigger↔ Interaction=QueryTriggerInteraction.UseGlobal, PreProcessing↔ Delegate preProcessRoots=null)

Performs a lag-compensated raycast query against all registered hitboxes. If the [HitOptions.IncludePhysX](#) or [HitOptions.IncludeBox2D](#) flag is indicated, query will also include static colliders, Unity colliders are recommended for static geometry, rather than Hitboxes. Important: results are NOT sorted by distance.
- int [RaycastAll](#) (Vector3 origin, Vector3 direction, float length, [PlayerRef](#) player, List< [LagCompensatedHit](#) > hits, int layerMask=-1, bool clearHits=true, [HitOptions](#) options=[HitOptions.None](#), QueryTriggerInteraction queryTrigger↔ Interaction=QueryTriggerInteraction.UseGlobal, PreProcessingDelegate preProcessRoots=null)

Performs a lag-compensated raycast query against all registered hitboxes. If the [HitOptions.IncludePhysX](#) or [HitOptions.IncludeBox2D](#) flag is indicated, query will also include static colliders, Unity colliders are recommended for static geometry, rather than Hitboxes. Important: results are NOT sorted by distance.

Public Attributes

- int [BVHDepth](#)

Debug data from Broadphase BVH (tree depth).
- int [BVHNodes](#)

Debug data from Broadphase BVH (total nodes count).
- [LagCompensationDraw DrawInfo](#)

Debug data used to draw the BVH nodes and the lag compensation history.
- int [TotalHitboxes](#)

Debug data from lag compensation history (registered [Hitbox](#) count).

Additional Inherited Members

6.63.1 Detailed Description

Entry point for lag compensated [Hitbox](#) queries, which maintains a history buffer, and provides lag compensated raycast and overlap methods. Singleton instance is accessible through the property `Runner.LagCompensation`.

Usage - Call any of the following methods:

```
HitboxManager.Raycast ()
HitboxManager.RaycastAll ()
HitboxManager.PositionRotation ()
HitboxManager.OverlapSphere ()
```

These methods use the history buffer to perform a [Hitbox](#) query against a state consistent with how the indicated [PlayerRef](#) perceived them locally.

6.63.2 Member Function Documentation

6.63.2.1 GetPlayerTickAndAlpha()

```
void GetPlayerTickAndAlpha (
    PlayerRef player,
    out int? tickFrom,
    out int? tickTo,
    out float? alpha )
```

Gets the tick and alpha interpolate values for a player.

Parameters

<i>player</i>	The player reference.
<i>tickFrom</i>	The tick value from which to interpolate.
<i>tickTo</i>	The tick value to which to interpolate.
<i>alpha</i>	The interpolation alpha value.

6.63.2.2 GetStatisticsSnapshot()

```
LagCompensationStatisticsSnapshot GetStatisticsSnapshot ( )
```

Gets a snapshot of the lag compensation statistics.

Returns

The lag compensation statistics snapshot.

6.63.2.3 OverlapBox() [1/3]

```
int OverlapBox (
    BoxOverlapQuery query,
    List< LagCompensatedHit > hits,
    bool clearHits = true )
```

Performs a lag-compensated box overlap query against all registered hitboxes. If the [HitOptions.IncludePhysX](#) or [HitOptions.IncludeBox2D](#) flag is indicated, query will also include static colliders, Unity colliders are recommended for static geometry, rather than Hitboxes.

Parameters

<i>query</i>	The query containing all necessary information.
<i>hits</i>	List to be filled with hits (both hitboxes and/or static colliders, if included).
<i>clearHits</i>	Clear list of hits before filling with new ones (defaults to true).

Returns

The total number of hits found.

6.63.2.4 OverlapBox() [2/3]

```
int OverlapBox (
    Vector3 center,
    Vector3 extents,
    Quaternion orientation,
    int tick,
    int? tickTo,
    float? alpha,
    List< LagCompensatedHit > hits,
    int layerMask = -1,
    HitOptions options = HitOptions.None,
    bool clearHits = true,
    QueryTriggerInteraction queryTriggerInteraction = QueryTriggerInteraction.UseGlobal,
    PreProcessingDelegate preProcessRoots = null )
```

Performs a lag-compensated box overlap query against all registered hitboxes. If the [HitOptions.IncludePhysX](#) or [HitOptions.IncludeBox2D](#) flag is indicated, query will also include static colliders, Unity colliders are recommended for static geometry, rather than Hitboxes.

Parameters

<i>center</i>	Center of the box in world space.
<i>extents</i>	Half of the size of the box in each dimension.
<i>orientation</i>	Rotation of the box.
<i>tick</i>	The exact tick to be queried
<i>tickTo</i>	Simulation tick number to use as the time reference for the lag compensation. If provided, must be combined with the <i>alpha</i> parameter for interpolation between <i>tick</i> and <i>tickTo</i> . If HitOptions.SubtickAccuracy is included on <i>options</i> , this query will be resolved against hitbox colliders interpolated between the specified ticks. Otherwise, only one of the two ticks will be considered, according to the rounded value of <i>alpha</i> .
<i>alpha</i>	Interpolation value when querying between <i>tick</i> and <i>tickTo</i> . If HitOptions.SubtickAccuracy is included on <i>options</i> , this query will be resolved against hitbox colliders interpolated between the specified ticks. Otherwise, only one of the two ticks will be considered, according to the rounded alpha value.
<i>hits</i>	List to be filled with hits (both hitboxes and/or static colliders, if included).
<i>layerMask</i>	Only objects with matching layers will be checked against.
<i>options</i>	Opt-in flags to compute with sub-tick accuracy (HitOptions.SubtickAccuracy) and/or to include PhysX (HitOptions.IncludePhysX) or Box2D (HitOptions.IncludeBox2D).
<i>clearHits</i>	Clear list of hits before filling with new ones (defaults to true).
<i>queryTriggerInteraction</i>	Trigger interaction behavior when also querying PhysX.
<i>preProcessRoots</i>	Delegate to pre-process HitboxRoots found in the broad-phase resolution of the query. Roots removed from the list will not be processed any further. Roots that remain on the candidates collection will be normally processed and fitting colliders will be evaluated in the query narrow-phase resolution. Hitbox collider indices added to the processed set will be evaluated in the narrow-phase regardless of further root processing steps (e.g. layer mask match).

Returns

The total number of hits found.

6.63.2.5 OverlapBox() [3/3]

```
int OverlapBox (
    Vector3 center,
    Vector3 extents,
    Quaternion orientation,
    PlayerRef player,
    List< LagCompensatedHit > hits,
    int layerMask = -1,
    HitOptions options = HitOptions.None,
    bool clearHits = true,
    QueryTriggerInteraction queryTriggerInteraction = QueryTriggerInteraction.UseGlobal,
    PreProcessingDelegate preProcessRoots = null )
```

Performs a lag-compensated box overlap query against all registered hitboxes. If the [HitOptions.IncludePhysX](#) or [HitOptions.IncludeBox2D](#) flag is indicated, query will also include static colliders, Unity colliders are recommended for static geometry, rather than Hitboxes.

Parameters

<i>center</i>	Center of the box in world space.
<i>extents</i>	Half of the size of the box in each dimension.
<i>orientation</i>	Rotation of the box.
<i>player</i>	Player who "owns" this overlap. Used by the server to find the exact hitbox snapshots to check against.
<i>hits</i>	List to be filled with hits (both hitboxes and/or static colliders, if included).
<i>layerMask</i>	Only objects with matching layers will be checked against.
<i>options</i>	Opt-in flags to compute with sub-tick accuracy (HitOptions.SubtickAccuracy) and/or to include PhysX (HitOptions.IncludePhysX) or Box2D (HitOptions.IncludeBox2D).
<i>clearHits</i>	Clear list of hits before filling with new ones (defaults to true).
<i>queryTriggerInteraction</i>	Trigger interaction behavior when also querying PhysX.
<i>preProcessRoots</i>	Delegate to pre-process HitboxRoots found in the broad-phase resolution of the query. Roots removed from the list will not be processed any further. Roots that remain on the candidates collection will be normally processed and fitting colliders will be evaluated in the query narrow-phase resolution. Hitbox collider indices added to the processed set will be evaluated in the narrow-phase regardless of further root processing steps (e.g. layer mask match).

Returns

The total number of hits found.

6.63.2.6 OverlapSphere() [1/3]

```
int OverlapSphere (
    SphereOverlapQuery query,
    List< LagCompensatedHit > hits,
    bool clearHits = true )
```

Performs a lag-compensated sphere overlap query against all registered hitboxes. If the [HitOptions.IncludePhysX](#) or [HitOptions.IncludeBox2D](#) flag is indicated, query will also include static colliders, Unity colliders are recommended for static geometry, rather than Hitboxes.

Parameters

<i>query</i>	The query containing all necessary information.
<i>hits</i>	List to be filled with hits (both hitboxes and/or static colliders, if included).
<i>clearHits</i>	Clear list of hits before filling with new ones (defaults to true).

Returns

The total number of hits found.

6.63.2.7 OverlapSphere() [2/3]

```
int OverlapSphere (
    Vector3 origin,
    float radius,
    int tick,
    int? tickTo,
    float? alpha,
    List< LagCompensatedHit > hits,
    int layerMask = -1,
    HitOptions options = HitOptions.None,
    bool clearHits = true,
    QueryTriggerInteraction queryTriggerInteraction = QueryTriggerInteraction.UseGlobal,
    PreProcessingDelegate preProcessRoots = null )
```

Performs a lag-compensated overlap sphere query against all registered hitboxes. If the [HitOptions.IncludePhysX](#) or [HitOptions.IncludeBox2D](#) flag is indicated, query will also include static colliders, Unity colliders are recommended for static geometry, rather than Hitboxes.

Parameters

<i>origin</i>	Sphere center, in world-space
<i>radius</i>	Sphere radius
<i>tick</i>	The tick to be queried
<i>tickTo</i>	Simulation tick number to use as the time reference for the lag compensation. If provided, must be combined with the <i>alpha</i> parameter for interpolation between <i>tick</i> and <i>tickTo</i> . If HitOptions.SubtickAccuracy is included on <i>options</i> , this query will be resolved against hitbox colliders interpolated between the specified ticks. Otherwise, only one of the two ticks will be considered, according to the rounded value of <i>alpha</i> .

Parameters

<i>alpha</i>	Interpolation value when querying between <i>tick</i> and <i>tickTo</i> . If HitOptions.SubtickAccuracy is included on <i>options</i> , this query will be resolved against hitbox colliders interpolated between the specified ticks. Otherwise, only one of the two ticks will be considered, according to the rounded alpha value.
<i>hits</i>	List to be filled with hits (both hitboxes and/or static colliders, if included).
<i>layerMask</i>	Only objects with matching layers will be checked against.
<i>options</i>	Opt-in flags to compute with sub-tick accuracy (HitOptions.SubtickAccuracy) and/or to include PhysX (HitOptions.IncludePhysX) or Box2D (HitOptions.IncludeBox2D).
<i>clearHits</i>	Clear list of hits before filling with new ones (defaults to true).
<i>queryTriggerInteraction</i>	Trigger interaction behavior when also querying PhysX.
<i>preProcessRoots</i>	Delegate to pre-process HitboxRoots found in the broad-phase resolution of the query. Roots removed from the list will not be processed any further. Roots that remain on the candidates collection will be normally processed and fitting colliders will be evaluated in the query narrow-phase resolution. Hitbox collider indices added to the processed set will be evaluated in the narrow-phase regardless of further root processing steps (e.g. layer mask match).

Returns

total number of hits

6.63.2.8 [OverlapSphere\(\)](#) [3/3]

```
int OverlapSphere (
    Vector3 origin,
    float radius,
    PlayerRef player,
    List< LagCompensatedHit > hits,
    int layerMask = -1,
    HitOptions options = HitOptions.None,
    bool clearHits = true,
    QueryTriggerInteraction queryTriggerInteraction = QueryTriggerInteraction.UseGlobal,
    PreProcessingDelegate preProcessRoots = null )
```

Performs a lag-compensated overlap sphere query against all registered hitboxes. If the [HitOptions.IncludePhysX](#) or [HitOptions.IncludeBox2D](#) flag is indicated, query will also include static colliders, Unity colliders are recommended for static geometry, rather than Hitboxes.

Parameters

<i>origin</i>	Sphere center, in world-space
<i>radius</i>	Sphere radius
<i>player</i>	Player who "owns" this overlap. Used by the server to find the exact hitbox snapshots to check against.
<i>hits</i>	List to be filled with hits (both hitboxes and/or static colliders, if included).
<i>layerMask</i>	Only objects with matching layers will be checked against.
<i>options</i>	Opt-in flags to compute with sub-tick accuracy (HitOptions.SubtickAccuracy) and/or to include PhysX (HitOptions.IncludePhysX) or Box2D (HitOptions.IncludeBox2D).

Parameters

<i>clearHits</i>	Clear list of hits before filling with new ones (defaults to true).
<i>queryTriggerInteraction</i>	Trigger interaction behavior when also querying PhysX.
<i>preProcessRoots</i>	Delegate to pre-process HitboxRoots found in the broad-phase resolution of the query. Roots removed from the list will not be processed any further. Roots that remain on the candidates collection will be normally processed and fitting colliders will be evaluated in the query narrow-phase resolution. Hitbox collider indices added to the processed set will be evaluated in the narrow-phase regardless of further root processing steps (e.g. layer mask match).

Returns

total number of hits

6.63.2.9 PositionRotation() [1/2]

```
void PositionRotation (
    Hitbox hitbox,
    int tick,
    out Vector3 position,
    out Quaternion rotation,
    bool subtickAccuracy = false,
    int? tickTo = null,
    float? alpha = null )
```

Performs a lag-compensated query for a specific [Hitbox](#) position and rotation.

Parameters

<i>hitbox</i>	The target hitbox to be queried in the past
<i>tick</i>	The tick to be queried
<i>tickTo</i>	Simulation tick number to use as the time reference for the lag compensation. If provided, must be combined with the <i>alpha</i> parameter for interpolation between <i>tick</i> and <i>tickTo</i> . If <i>subtickAccuracy</i> is requested, the query will return the hitbox state interpolated between the specified ticks. Otherwise, only one of the two ticks will be considered, according to the rounded value of <i>alpha</i> .
<i>alpha</i>	Interpolation value when querying between <i>tick</i> and <i>tickTo</i> . If <i>subtickAccuracy</i> is requested, the query will return the hitbox state interpolated between the specified ticks. Otherwise, only one of the two ticks will be considered, according to the rounded alpha value.
<i>position</i>	Will be filled with the hitbox position at the time of the tick
<i>rotation</i>	Will be filled with the hitbox rotation at the time of the tick
<i>subtickAccuracy</i>	If the query should interpolate between ticks to reflect exactly what was seen on the client.

6.63.2.10 PositionRotation() [2/2]

```
void PositionRotation (
```

```

Hitbox hitbox,
PlayerRef player,
out Vector3 position,
out Quaternion rotation,
bool subTickAccuracy = false )

```

Performs a lag-compensated query for a specific [Hitbox](#) position and rotation.

Parameters

<i>hitbox</i>	The target hitbox to be queried in the past
<i>player</i>	Player who "owns" this overlap. Used by the server to find the exact hitbox snapshots to check against.
<i>position</i>	Will be filled with the hitbox position at the time of the tick
<i>rotation</i>	Will be filled with the hitbox rotation at the time of the tick
<i>subTickAccuracy</i>	If the query should interpolate between ticks to reflect exactly what was seen on the client.

6.63.2.11 Raycast() [1/3]

```

bool Raycast (
    RaycastQuery query,
    out LagCompensatedHit hit )

```

Performs a lag-compensated raycast query against all registered hitboxes. If the [HitOptions.IncludePhysX](#) or [HitOptions.IncludeBox2D](#) flag is indicated, query will also include static colliders, Unity colliders are recommended for static geometry, rather than Hitboxes.

Parameters

<i>query</i>	The query containing all necessary information.
<i>hit</i>	Raycast results will be filled in here.

Returns

The total number of hits found.

6.63.2.12 Raycast() [2/3]

```

bool Raycast (
    Vector3 origin,
    Vector3 direction,
    float length,
    int tick,
    int? tickTo,
    float? alpha,
    out LagCompensatedHit hit,

```

```
int layerMask = -1,
HitOptions options = HitOptions.None,
QueryTriggerInteraction queryTriggerInteraction = QueryTriggerInteraction.UseGlobal,
PreProcessingDelegate preProcessRoots = null )
```

Performs a lag-compensated raycast query against all registered hitboxes. If the [HitOptions.IncludePhysX](#) or [HitOptions.IncludeBox2D](#) flag is indicated, query will also include static colliders, Unity colliders are recommended for static geometry, rather than Hitboxes.

Parameters

<i>origin</i>	Raycast origin, in world-space
<i>direction</i>	Raycast direction, in world-space
<i>length</i>	Raycast length
<i>tick</i>	Simulation tick number to use as the time reference for the lag compensation (use this for server AI, and similar).
<i>tickTo</i>	Simulation tick number to use as the time reference for the lag compensation. If provided, must be combined with the <i>alpha</i> parameter for interpolation between <i>tick</i> and <i>tickTo</i> . If HitOptions.SubtickAccuracy is included on <i>options</i> , this query will be resolved against hitbox colliders interpolated between the specified ticks. Otherwise, only one of the two ticks will be considered, according to the rounded value of <i>alpha</i> .
<i>alpha</i>	Interpolation value when querying between <i>tick</i> and <i>tickTo</i> . If HitOptions.SubtickAccuracy is included on <i>options</i> , this query will be resolved against hitbox colliders interpolated between the specified ticks. Otherwise, only one of the two ticks will be considered, according to the rounded alpha value.
<i>hit</i>	Raycast results will be filled in here.
<i>layerMask</i>	Only objects with matching layers will be checked against.
<i>options</i>	Opt-in flags to compute with sub-tick accuracy (HitOptions.SubtickAccuracy) and/or to include PhysX (HitOptions.IncludePhysX) or Box2D (HitOptions.IncludeBox2D).
<i>queryTriggerInteraction</i>	Trigger interaction behavior when also querying PhysX.
<i>preProcessRoots</i>	Delegate to pre-process HitboxRoots found in the broad-phase resolution of the query. Roots removed from the list will not be processed any further. Roots that remain on the candidates collection will be normally processed and fitting colliders will be evaluated in the query narrow-phase resolution. Hitbox collider indices added to the processed set will be evaluated in the narrow-phase regardless of further root processing steps (e.g. layer mask match).

Returns

True if something is hit

6.63.2.13 Raycast() [3/3]

```
bool Raycast (
    Vector3 origin,
    Vector3 direction,
    float length,
    PlayerRef player,
    out LagCompensatedHit hit,
    int layerMask = -1,
```

```

HitOptions options = HitOptions.None,
QueryTriggerInteraction queryTriggerInteraction = QueryTriggerInteraction.UseGlobal,
PreProcessingDelegate preProcessRoots = null )

```

Performs a lag-compensated raycast query against all registered hitboxes. If the [HitOptions.IncludePhysX](#) or [HitOptions.IncludeBox2D](#) flag is indicated, query will also include static colliders, Unity colliders are recommended for static geometry, rather than Hitboxes.

Parameters

<i>origin</i>	Raycast origin, in world-space
<i>direction</i>	Raycast direction, in world-space
<i>length</i>	Raycast length
<i>player</i>	Player who "owns" this raycast. Used by the server to find the exact hitbox snapshots to check against.
<i>hit</i>	Raycast results will be filled in here.
<i>layerMask</i>	Only objects with matching layers will be checked against.
<i>options</i>	Opt-in flags to compute with sub-tick accuracy (HitOptions.SubtickAccuracy) and/or to include PhysX (HitOptions.IncludePhysX) or Box2D (HitOptions.IncludeBox2D).
<i>queryTriggerInteraction</i>	Trigger interaction behavior when also querying PhysX.
<i>preProcessRoots</i>	Delegate to pre-process HitboxRoots found in the broad-phase resolution of the query. Roots removed from the list will not be processed any further. Roots that remain on the candidates collection will be normally processed and fitting colliders will be evaluated in the query narrow-phase resolution. Hitbox collider indices added to the processed set will be evaluated in the narrow-phase regardless of further root processing steps (e.g. layer mask match).

Returns

True if something is hit

6.63.2.14 RaycastAll() [1/3]

```

int RaycastAll (
    RaycastAllQuery query,
    List< LagCompensatedHit > hits,
    bool clearHits = true )

```

Performs a lag-compensated raycast query against all registered hitboxes. If the [HitOptions.IncludePhysX](#) or [HitOptions.IncludeBox2D](#) flag is indicated, query will also include static colliders, Unity colliders are recommended for static geometry, rather than Hitboxes.

Parameters

<i>query</i>	The query containing all necessary information.
<i>hits</i>	List to be filled with hits (both hitboxes and/or static colliders, if included).
<i>clearHits</i>	Clear list of hits before filling with new ones (defaults to true).

Returns

The total number of hits found.

6.63.2.15 RaycastAll() [2/3]

```
int RaycastAll (
    Vector3 origin,
    Vector3 direction,
    float length,
    int tick,
    int? tickTo,
    float? alpha,
    List< LagCompensatedHit > hits,
    int layerMask = -1,
    bool clearHits = true,
    HitOptions options = HitOptions.None,
    QueryTriggerInteraction queryTriggerInteraction = QueryTriggerInteraction.UseGlobal,
    PreProcessingDelegate preProcessRoots = null )
```

Performs a lag-compensated raycast query against all registered hitboxes. If the [HitOptions.IncludePhysX](#) or [HitOptions.IncludeBox2D](#) flag is indicated, query will also include static colliders, Unity colliders are recommended for static geometry, rather than Hitboxes. Important: results are NOT sorted by distance.

Parameters

<i>origin</i>	Raycast origin, in world-space
<i>direction</i>	Raycast direction, in world-space
<i>length</i>	Raycast length
<i>tick</i>	Simulation tick number to use as the time reference for the lag compensation (use this for server AI, and similar).
<i>tickTo</i>	Simulation tick number to use as the time reference for the lag compensation. If provided, must be combined with the <i>alpha</i> parameter for interpolation between <i>tick</i> and <i>tickTo</i> . If HitOptions.SubtickAccuracy is included on <i>options</i> , this query will be resolved against hitbox colliders interpolated between the specified ticks. Otherwise, only one of the two ticks will be considered, according to the rounded value of <i>alpha</i> .
<i>alpha</i>	Interpolation value when querying between <i>tick</i> and <i>tickTo</i> . If HitOptions.SubtickAccuracy is included on <i>options</i> , this query will be resolved against hitbox colliders interpolated between the specified ticks. Otherwise, only one of the two ticks will be considered, according to the rounded alpha value.
<i>hits</i>	List to be filled with hits (both hitboxes and/or static colliders, if included).
<i>layerMask</i>	Only objects with matching layers will be checked against.
<i>options</i>	Opt-in flags to compute with sub-tick accuracy (HitOptions.SubtickAccuracy) and/or to include PhysX (HitOptions.IncludePhysX) or Box2D (HitOptions.IncludeBox2D).
<i>clearHits</i>	Clear list of hits before filling with new ones (defaults to true).
<i>queryTriggerInteraction</i>	Trigger interaction behavior when also querying PhysX.
<i>preProcessRoots</i>	Delegate to pre-process HitboxRoots found in the broad-phase resolution of the query. Roots removed from the list will not be processed any further. Roots that remain on the candidates collection will be normally processed and fitting colliders will be evaluated in the query narrow-phase resolution. Hitbox collider indices added to the processed set will be evaluated in the narrow-phase regardless of further root processing steps (e.g. layer mask match).

Returns

total number of hits

6.63.2.16 RaycastAll() [3/3]

```
int RaycastAll (
    Vector3 origin,
    Vector3 direction,
    float length,
    PlayerRef player,
    List< LagCompensatedHit > hits,
    int layerMask = -1,
    bool clearHits = true,
    HitOptions options = HitOptions.None,
    QueryTriggerInteraction queryTriggerInteraction = QueryTriggerInteraction.UseGlobal,
    PreProcessingDelegate preProcessRoots = null )
```

Performs a lag-compensated raycast query against all registered hitboxes. If the [HitOptions.IncludePhysX](#) or [HitOptions.IncludeBox2D](#) flag is indicated, query will also include static colliders, Unity colliders are recommended for static geometry, rather than Hitboxes. Important: results are NOT sorted by distance.

Parameters

<i>origin</i>	Raycast origin, in world-space
<i>direction</i>	Raycast direction, in world-space
<i>length</i>	Raycast length
<i>player</i>	Player who "owns" this raycast. Used by the server to find the exact hitbox snapshots to check against.
<i>hits</i>	List to be filled with hits (both hitboxes and/or static colliders, if included).
<i>layerMask</i>	Only objects with matching layers will be checked against.
<i>options</i>	Opt-in flags to compute with sub-tick accuracy (HitOptions.SubtickAccuracy) and/or to include PhysX (HitOptions.IncludePhysX) or Box2D (HitOptions.IncludeBox2D).
<i>clearHits</i>	Clear list of hits before filling with new ones (defaults to true).
<i>queryTriggerInteraction</i>	Trigger interaction behavior when also querying PhysX.
<i>preProcessRoots</i>	Delegate to pre-process HitboxRoots found in the broad-phase resolution of the query. Roots removed from the list will not be processed any further. Roots that remain on the candidates collection will be normally processed and fitting colliders will be evaluated in the query narrow-phase resolution. Hitbox collider indices added to the processed set will be evaluated in the narrow-phase regardless of further root processing steps (e.g. layer mask match).

Returns

total number of hits

6.63.3 Member Data Documentation

6.63.3.1 BVHDepth

`int BVHDepth`

Debug data from Broadphase BVH (tree depth).

6.63.3.2 BVHNodes

`int BVHNodes`

Debug data from Broadphase BVH (total nodes count).

6.63.3.3 DrawInfo

`LagCompensationDraw DrawInfo`

Debug data used to draw the BVH nodes and the lag compensation history.

6.63.3.4 TotalHitboxes

`int TotalHitboxes`

Debug data from lag compensation history (registered [Hitbox](#) count).

6.64 HitboxRoot Class Reference

Root [Hitbox](#) group container. Manages registering/unregistering hitboxes with the group, and defines the broad-phase geometry for the group.

Inherits [NetworkBehaviour](#).

Public Types

- enum class [ConfigFlags](#) : int
Set of configuration options for a [Hitbox](#) Root behaviour.

Public Member Functions

- override void [Despawned](#) ([NetworkRunner](#) runner, bool hasState)
Called before the network object is despawned
- void [InitHitboxes](#) ()
Finds child [Hitbox](#) components, and adds them to the [Hitboxes](#) collection.
- bool [IsHitboxActive](#) ([Hitbox](#) hitbox)
Checks the state of a [Hitbox](#) instance under this root. Both the hitbox and its root must be active in order for it to be hit by lag-compensated queries.
- void [OnDrawGizmos](#) ()
[HitboxRoot](#) on draw gizmos.
- void [SetHitboxActive](#) ([Hitbox](#) hitbox, bool setActive)
Sets the state of a [Hitbox](#) instance under this root. Both the hitbox and its root must be active in order for it to be hit by lag-compensated queries.
- void [SetMinBoundingRadius](#) ()
Sets [BroadRadius](#) to a rough value which encompasses all [Hitboxes](#) in their current positions.

Public Attributes

- float [BroadRadius](#)
The radius of the broadphase bounding sphere for this [Hitbox](#) group. Used by [HitboxManager](#) to insert/update lag compensated [NetworkObjects](#) into its BVH (bounding volume hierarchy) data structure. Be sure this radius encompasses all children [Hitbox](#) components (including their full ranges of animation motion). We plan to offer an option to dynamically compute the bounding volume, but the performance trade of will still favor a hand-crafted radius.
- [ConfigFlags](#) [Config](#) = [ConfigFlags.Default](#)
Set of configuration options for this [Hitbox](#) Root behaviour. Check the API documentation for more details on what each flag represents.
- Color [GizmosColor](#) = Color.gray
Color used when drawing gizmos for this hitbox.
- [Hitbox](#)[] [Hitboxes](#)
All [Hitbox](#) instances in hierarchy. Auto-filled at Spawnd.
- Vector3 [Offset](#)
Local-space offset of the broadphase bounding sphere from its transform position.

Static Public Attributes

- const Int32 [MAX_HITBOXES](#) = (sizeof(UInt32) * 8) - 1
The max number of hitboxes allowed under the same root.

Protected Member Functions

- virtual void [DrawGizmos](#) (Color color, ref Matrix4x4 localToWorldMatrix)
Draws the gizmos for the [HitboxRoot](#)

Properties

- bool [HitboxRootActive](#) [get, set]
Get or set the state of this [HitboxRoot](#). For a hitbox to be hit by lag-compensated queries, both it and its [HitboxRoot](#) must be active.
- bool [InInterest](#) [get]
If this [HitboxRoot](#) is in interest for the local player.
- [HitboxManager](#) [Manager](#) [get]
Reference to associated hitbox manager (from which lag compensated queries can be performed).

Additional Inherited Members

6.64.1 Detailed Description

Root [Hitbox](#) group container. Manages registering/unregistering hitboxes with the group, and defines the broad-phase geometry for the group.

Broadphase is the initial rough query used by raycasts/overlaps/etc to find potential hit candidates, which are then used in the final **narrowphase** query.

6.64.2 Member Enumeration Documentation

6.64.2.1 ConfigFlags

```
enum ConfigFlags : int [strong]
```

Set of configuration options for a [Hitbox](#) Root behaviour.

Enumerator

ReinitializeHitboxesBeforeRegistration	If the collection of hitboxes under a given root should be re-initialized before the Root is registered in a hitbox snapshot. If disabled, the hitboxes will be used as configured in edit-time.
IncludeInactiveHitboxes	If Hitboxes on inactive Game Objects should be registered under this root upon initialization.
Legacy	Set of configuration flags that replicate the behaviour as it was before the flag options were added.
Default	Set of configuration flags with the default behaviour, suitable for most use-cases.

6.64.3 Member Function Documentation

6.64.3.1 DrawGizmos()

```
virtual void DrawGizmos (
    Color color,
    ref Matrix4x4 localToWorldMatrix ) [protected], [virtual]
```

Draws the gizmos for the [HitboxRoot](#)

6.64.3.2 InitHitboxes()

```
void InitHitboxes ( )
```

Finds child [Hitbox](#) components, and adds them to the [Hitboxes](#) collection.

6.64.3.3 IsHitboxActive()

```
bool IsHitboxActive (
    Hitbox hitbox )
```

Checks the state of a [Hitbox](#) instance under this root. Both the hitbox and its root must be active in order for it to be hit by lag-compensated queries.

Parameters

<i>hitbox</i>	A hitbox instance under the hierarchy of this root.
---------------	---

Returns

True if the *hitbox* is part of this root and is active.

Exceptions

<i>ArgumentOutOfRangeException</i>	If the Hitbox.HitboxIndex of the <i>hitbox</i> is outside the valid range.
<i>AssertException</i>	In Debug configuration, if the <i>hitbox</i> is not part of this root.

6.64.3.4 OnDrawGizmos()

```
void OnDrawGizmos ( )
```

[HitboxRoot](#) on draw gizmos.

6.64.3.5 SetHitboxActive()

```
void SetHitboxActive (
    Hitbox hitbox,
    bool setActive )
```

Sets the state of a [Hitbox](#) instance under this root. Both the hitbox and its root must be active in order for it to be hit by lag-compensated queries.

Parameters

<i>hitbox</i>	A hitbox instance under the hierarchy of this root.
<i>setActive</i>	If the hitbox should be activated or deactivated.

Exceptions

<i>ArgumentOutOfRangeException</i>	If the Hitbox.HitboxIndex of the <i>hitbox</i> is outside the valid range.
<i>AssertException</i>	In Debug configuration, if the <i>hitbox</i> is not part of this root.

6.64.3.6 SetMinBoundingRadius()

```
void SetMinBoundingRadius ( )
```

Sets [BroadRadius](#) to a rough value which encompasses all [Hitboxes](#) in their current positions.

6.64.4 Member Data Documentation**6.64.4.1 BroadRadius**

```
float BroadRadius
```

The radius of the broadphase bounding sphere for this [Hitbox](#) group. Used by [HitboxManager](#) to insert/update lag compensated NetworkObjects into its BVH (bounding volume hierarchy) data structure. Be sure this radius encompasses all children [Hitbox](#) components (including their full ranges of animation motion). We plan to offer an option to dynamically compute the bounding volume, but the performance trade of will still favor a hand-crafted radius.

Broadphase is the initial rough query used by raycasts/overlaps/etc to find potential hit candidates, which are then used in the final **narrowphase** query.

6.64.4.2 Config

```
ConfigFlags Config = ConfigFlags.Default
```

Set of configuration options for this [Hitbox](#) Root behaviour. Check the API documentation for more details on what each flag represents.

6.64.4.3 GizmosColor

```
Color GizmosColor = Color.gray
```

Color used when drawing gizmos for this hitbox.

6.64.4.4 Hitboxes

```
Hitbox [] Hitboxes
```

All [Hitbox](#) instances in hierarchy. Auto-filled at Spawned.

6.64.4.5 MAX_HITBOXES

```
const Int32 MAX_HITBOXES = (sizeof(UInt32) * 8) - 1 [static]
```

The max number of hitboxes allowed under the same root.

6.64.4.6 Offset

```
Vector3 Offset
```

Local-space offset of the broadphase bounding sphere from its transform position.

Adjust the [BroadRadius](#) and [Offset](#) until the sphere gizmo (shown in the Unity Scene window) encompasses all children [Hitbox](#) components (including their full ranges of animation motion).

Broadphase is the initial rough query used by raycasts/overlaps/etc to find potential hit candidates, which are then used in the final **narrowphase** query.

6.64.5 Property Documentation

6.64.5.1 HitboxRootActive

```
bool HitboxRootActive [get], [set]
```

Get or set the state of this [HitboxRoot](#). For a hitbox to be hit by lag-compensated queries, both it and its [HitboxRoot](#) must be active.

6.64.5.2 InInterest

```
bool InInterest [get]
```

If this [HitboxRoot](#) is in interest for the local player.

6.64.5.3 Manager

```
HitboxManager Manager [get]
```

Reference to associated hitbox manager (from which lag compensated queries can be performed).

6.65 HostMigrationConfig Class Reference

Project configuration settings specific to how the Host Migration behaves.

Public Attributes

- bool [EnableAutoUpdate](#)
Enabled the Host Migration feature
- int [UpdateDelay](#) = 10
Delay between Host Migration Snapshot updates

6.65.1 Detailed Description

Project configuration settings specific to how the Host Migration behaves.

6.65.2 Member Data Documentation

6.65.2.1 EnableAutoUpdate

```
bool EnableAutoUpdate
```

Enabled the Host Migration feature

6.65.2.2 UpdateDelay

```
int UpdateDelay = 10
```

Delay between Host Migration Snapshot updates

6.66 HostMigrationToken Class Reference

Transitory Holder with all necessary information to restart the [Fusion](#) Runner after the Host Migration has completed

Properties

- [GameMode](#) [GameMode](#) [get]
New GameMode the local peer will assume after the Host Migration

6.66.1 Detailed Description

Transitory Holder with all necessary information to restart the [Fusion](#) Runner after the Host Migration has completed

6.66.2 Property Documentation

6.66.2.1 GameMode

[GameMode](#) [GameMode](#) [get]

New GameMode the local peer will assume after the Host Migration

6.67 IAfterAllTicks Interface Reference

Interface for [AfterAllTicks](#) callback. Called after the re-simulation loop (when applicable), and also after the forward simulation loop. Implement this interface on [SimulationBehaviour](#) and [NetworkBehaviour](#) classes.

Inherits [IPublicFacingInterface](#).

Inherited by [NetworkMecanimAnimator](#), and [NetworkTransform](#).

Public Member Functions

- void [AfterAllTicks](#) (bool resimulation, int tickCount)
Called after the re-simulation loop (when applicable), and also after the forward simulation loop. Only called on Updates where re-simulation or forward ticks are processed.

6.67.1 Detailed Description

Interface for [AfterAllTicks](#) callback. Called after the re-simulation loop (when applicable), and also after the forward simulation loop. Implement this interface on [SimulationBehaviour](#) and [NetworkBehaviour](#) classes.

6.67.2 Member Function Documentation

6.67.2.1 AfterAllTicks()

```
void AfterAllTicks (
    bool resimulation,
    int tickCount )
```

Called after the re-simulation loop (when applicable), and also after the forward simulation loop. Only called on Updates where re-simulation or forward ticks are processed.

Parameters

<i>resimulation</i>	True if this is being called during the re-simulation loop. False if during the forward simulation loop.
<i>tickCount</i>	How many re-simulation or forward ticks are going to be processed.

6.68 IAfterClientPredictionReset Interface Reference

Callback interface for [AfterClientPredictionReset](#). Called at the very start of the resimulation loop (on clients with prediction enabled), immediately after state is set to the latest server snapshot. Implement this interface on [SimulationBehaviour](#) and [NetworkBehaviour](#) classes.

Inherits [IPublicFacingInterface](#).

Public Member Functions

- void [AfterClientPredictionReset](#) ()

Called at the very start of the resimulation loop (on clients with prediction enabled), immediately after state is set to the latest server snapshot.

6.68.1 Detailed Description

Callback interface for [AfterClientPredictionReset](#). Called at the very start of the resimulation loop (on clients with prediction enabled), immediately after state is set to the latest server snapshot. Implement this interface on [SimulationBehaviour](#) and [NetworkBehaviour](#) classes.

6.68.2 Member Function Documentation

6.68.2.1 AfterClientPredictionReset()

```
void AfterClientPredictionReset ( )
```

Called at the very start of the resimulation loop (on clients with prediction enabled), immediately after state is set to the latest server snapshot.

6.69 IAfterHostMigration Interface Reference

Used to mark NetworkBehaviors that need to be react after a Host Migration process

Inherits [IPublicFacingInterface](#).

Public Member Functions

- void [AfterHostMigration](#) ()

Invoked after the Host Migration happens in order to setup non-networked data on NetworkBehaviors

6.69.1 Detailed Description

Used to mark NetworkBehaviors that need to be react after a Host Migration process

6.69.2 Member Function Documentation

6.69.2.1 AfterHostMigration()

```
void AfterHostMigration ( )
```

Invoked after the Host Migration happens in order to setup non-networked data on NetworkBehaviors

6.70 IAfterRender Interface Reference

Interface for [AfterRender](#) callback. Called after the render loop.

Inherits [IPublicFacingInterface](#).

Public Member Functions

- void [AfterRender](#) ()

Called after the render loop.

6.70.1 Detailed Description

Interface for [AfterRender](#) callback. Called after the render loop.

6.70.2 Member Function Documentation

6.70.2.1 AfterRender()

```
void AfterRender ( )
```

Called after the render loop.

6.71 IAfterSpawned Interface Reference

Interface for [AfterSpawned](#) callback. Called after the object is spawned.

Inherits [IPublicFacingInterface](#).

Public Member Functions

- void [AfterSpawned](#) ()
Called after the object is spawned.

6.71.1 Detailed Description

Interface for [AfterSpawned](#) callback. Called after the object is spawned.

6.71.2 Member Function Documentation

6.71.2.1 AfterSpawned()

```
void AfterSpawned ( )
```

Called after the object is spawned.

6.72 IAfterTick Interface Reference

Interface for [AfterTick](#) callback. Called after each tick simulation completes. Implement this interface on [SimulationBehaviour](#) and [NetworkBehaviour](#) classes.

Inherits [IPublicFacingInterface](#).

Inherited by [HitboxManager](#).

Public Member Functions

- void [AfterTick](#) ()
Called after each tick simulation completes.

6.72.1 Detailed Description

Interface for [AfterTick](#) callback. Called after each tick simulation completes. Implement this interface on [SimulationBehaviour](#) and [NetworkBehaviour](#) classes.

6.72.2 Member Function Documentation

6.72.2.1 AfterTick()

```
void AfterTick ( )
```

Called after each tick simulation completes.

6.73 IAfterUpdate Interface Reference

Interface for the [AfterUpdate](#) callback, which is called at the end of each [Fusion](#) Update segment. Implement this interface on [SimulationBehaviour](#) and [NetworkBehaviour](#) classes.

Inherits [IPublicFacingInterface](#).

Public Member Functions

- void [AfterUpdate](#) ()

Called at the end of the [Fusion](#) Update loop, before all Unity MonoBehaviour.Update() callbacks.

6.73.1 Detailed Description

Interface for the [AfterUpdate](#) callback, which is called at the end of each [Fusion](#) Update segment. Implement this interface on [SimulationBehaviour](#) and [NetworkBehaviour](#) classes.

6.73.2 Member Function Documentation

6.73.2.1 AfterUpdate()

```
void AfterUpdate ( )
```

Called at the end of the [Fusion](#) Update loop, before all Unity MonoBehaviour.Update() callbacks.

6.74 IAfterUpdateRemotePrefabs Interface Reference

Invoked after updating remote prefabs

Inherits [IPublicFacingInterface](#).

Public Member Functions

- void [AfterUpdateRemotePrefabs](#) ()
Invoked after updating remote prefabs

6.74.1 Detailed Description

Invoked after updating remote prefabs

6.74.2 Member Function Documentation

6.74.2.1 AfterUpdateRemotePrefabs()

```
void AfterUpdateRemotePrefabs ( )
```

Invoked after updating remote prefabs

6.75 IAsyncOperation Interface Reference

Defines an asynchronous operation.

Inherited by [ICoroutine](#).

Properties

- ExceptionDispatchInfo [Error](#) [get]
Gets the exception information if an error occurred during the operation.
- bool [IsDone](#) [get]
Gets a value indicating whether the operation is done.

Events

- Action< [IAsyncOperation](#) > [Completed](#)
Occurs when the operation is completed.

6.75.1 Detailed Description

Defines an asynchronous operation.

6.75.2 Property Documentation

6.75.2.1 Error

```
ExceptionDispatchInfo Error [get]
```

Gets the exception information if an error occurred during the operation.

6.75.2.2 IsDone

```
bool IsDone [get]
```

Gets a value indicating whether the operation is done.

6.75.3 Event Documentation

6.75.3.1 Completed

```
Action<IAsyncOperation> Completed
```

Occurs when the operation is completed.

6.76 IBeforeAllTicks Interface Reference

Interface for [BeforeAllTicks](#) callback. Called before the re-simulation loop (when applicable), and also before the forward simulation loop. Implement this interface on [SimulationBehaviour](#) and [NetworkBehaviour](#) classes.

Inherits [IPublicFacingInterface](#).

Inherited by [NetworkTransform](#).

Public Member Functions

- void [BeforeAllTicks](#) (bool resimulation, int tickCount)

Called before the re-simulation loop (when applicable), and also before the forward simulation loop. Only called on Updates where re-simulation or forward ticks are processed.

6.76.1 Detailed Description

Interface for [BeforeAllTicks](#) callback. Called before the re-simulation loop (when applicable), and also before the forward simulation loop. Implement this interface on [SimulationBehaviour](#) and [NetworkBehaviour](#) classes.

6.76.2 Member Function Documentation

6.76.2.1 BeforeAllTicks()

```
void BeforeAllTicks (
    bool resimulation,
    int tickCount )
```

Called before the re-simulation loop (when applicable), and also before the forward simulation loop. Only called on Updates where re-simulation or forward ticks are processed.

Parameters

<i>resimulation</i>	True if this is being called during the re-simulation loop. False if during the forward simulation loop.
<i>tickCount</i>	How many re-simulation or forward ticks are going to be processed.

6.77 IBeforeClientPredictionReset Interface Reference

Callback interface for [BeforeClientPredictionReset](#). Called at the very start of the re-simulation loop (on clients with prediction enabled), before state is set to the latest server snapshot. Implement this interface on [SimulationBehaviour](#) and [NetworkBehaviour](#) classes.

Inherits [IPublicFacingInterface](#).

Public Member Functions

- void [BeforeClientPredictionReset](#) ()

Called at the very start of the re-simulation loop (on clients with prediction enabled), before state is set to the latest server snapshot.

6.77.1 Detailed Description

Callback interface for [BeforeClientPredictionReset](#). Called at the very start of the re-simulation loop (on clients with prediction enabled), before state is set to the latest server snapshot. Implement this interface on [SimulationBehaviour](#) and [NetworkBehaviour](#) classes.

6.77.2 Member Function Documentation

6.77.2.1 BeforeClientPredictionReset()

```
void BeforeClientPredictionReset ( )
```

Called at the very start of the re-simulation loop (on clients with prediction enabled), before state is set to the latest server snapshot.

6.78 IBeforeCopyPreviousState Interface Reference

Interface for [BeforeCopyPreviousState](#) callback. Called before the copy of the previous state.

Inherits [IPublicFacingInterface](#).

Inherited by [NetworkTransform](#).

Public Member Functions

- void [BeforeCopyPreviousState](#) ()
Called before the copy of the previous state.

6.78.1 Detailed Description

Interface for [BeforeCopyPreviousState](#) callback. Called before the copy of the previous state.

6.78.2 Member Function Documentation

6.78.2.1 BeforeCopyPreviousState()

```
void BeforeCopyPreviousState ( )
```

Called before the copy of the previous state.

6.79 IBeforeHitboxRegistration Interface Reference

Interface for [BeforeHitboxRegistration](#) callback. Implement this interface on [SimulationBehaviour](#) and [NetworkBehaviour](#) classes.

Inherits [IPublicFacingInterface](#).

Public Member Functions

- void [BeforeHitboxRegistration](#) ()
Called immediately before the [HitboxManager](#) registers hitboxes in a snapshot.

6.79.1 Detailed Description

Interface for [BeforeHitboxRegistration](#) callback. Implement this interface on [SimulationBehaviour](#) and [NetworkBehaviour](#) classes.

6.79.2 Member Function Documentation

6.79.2.1 BeforeHitboxRegistration()

```
void BeforeHitboxRegistration ( )
```

Called immediately before the [HitboxManager](#) registers hitboxes in a snapshot.

6.80 IBeforeSimulation Interface Reference

Interface for [BeforeSimulation](#) callback. Called before both the re-simulation (when applicable) and forward simulation loops. Implement this interface on [SimulationBehaviour](#) and [NetworkBehaviour](#) classes.

Inherits [IPublicFacingInterface](#).

Inherited by [HitboxManager](#).

Public Member Functions

- void [BeforeSimulation](#) (int forwardTickCount)

Called before both the re-simulation (when applicable) and forward simulation loops. Only called on updates that have forward ticks to process.

6.80.1 Detailed Description

Interface for [BeforeSimulation](#) callback. Called before both the re-simulation (when applicable) and forward simulation loops. Implement this interface on [SimulationBehaviour](#) and [NetworkBehaviour](#) classes.

6.80.2 Member Function Documentation

6.80.2.1 BeforeSimulation()

```
void BeforeSimulation (
    int forwardTickCount )
```

Called before both the re-simulation (when applicable) and forward simulation loops. Only called on updates that have forward ticks to process.

Parameters

<code>forwardTickCount</code>	How many forward ticks are going to be processed.
-------------------------------	---

6.81 IBeforeTick Interface Reference

Interface for [BeforeTick](#) callback. Called before each tick is simulated. Implement this interface on [SimulationBehaviour](#) and [NetworkBehaviour](#) classes.

Inherits [IPublicFacingInterface](#).

Public Member Functions

- void [BeforeTick](#) ()
Called before each tick is simulated.

6.81.1 Detailed Description

Interface for [BeforeTick](#) callback. Called before each tick is simulated. Implement this interface on [SimulationBehaviour](#) and [NetworkBehaviour](#) classes.

6.81.2 Member Function Documentation

6.81.2.1 BeforeTick()

```
void BeforeTick ( )
```

Called before each tick is simulated.

6.82 IBeforeUpdate Interface Reference

Interface for the [BeforeUpdate](#) callback, which is called at the beginning of each [Fusion](#) Update segment. Implement this interface on [SimulationBehaviour](#) and [NetworkBehaviour](#) classes.

Inherits [IPublicFacingInterface](#).

Public Member Functions

- void [BeforeUpdate](#) ()
Called at the start of the [Fusion](#) Update loop, before the [Fusion](#) simulation loop.

6.82.1 Detailed Description

Interface for the [BeforeUpdate](#) callback, which is called at the beginning of each [Fusion](#) Update segment. Implement this interface on [SimulationBehaviour](#) and [NetworkBehaviour](#) classes.

6.82.2 Member Function Documentation

6.82.2.1 BeforeUpdate()

```
void BeforeUpdate ( )
```

Called at the start of the [Fusion](#) Update loop, before the [Fusion](#) simulation loop.

6.83 IBeforeUpdateRemotePrefabs Interface Reference

Invoked before updating remote prefabs

Inherits [IPublicFacingInterface](#).

Public Member Functions

- void [BeforeUpdateRemotePrefabs](#) ()
Invoked before updating remote prefabs

6.83.1 Detailed Description

Invoked before updating remote prefabs

6.83.2 Member Function Documentation

6.83.2.1 BeforeUpdateRemotePrefabs()

```
void BeforeUpdateRemotePrefabs ( )
```

Invoked before updating remote prefabs

6.84 ICoroutine Interface Reference

Defines a coroutine.

Inherits [IAsyncOperation](#), and [IEnumerator](#).

Additional Inherited Members

6.84.1 Detailed Description

Defines a coroutine.

6.85 IDespawned Interface Reference

Interface for [Despawned](#) callback. Called when a [NetworkBehaviour](#) is despawned.

Inherits [IPublicFacingInterface](#).

Inherited by [NetworkBehaviour](#).

Public Member Functions

- void [Despawned](#) ([NetworkRunner](#) runner, bool hasState)
Called when a [NetworkBehaviour](#) is despawned.

6.85.1 Detailed Description

Interface for [Despawned](#) callback. Called when a [NetworkBehaviour](#) is despawned.

6.85.2 Member Function Documentation

6.85.2.1 Despawned()

```
void Despawned (
    NetworkRunner runner,
    bool hasState )
```

Called when a [NetworkBehaviour](#) is despawned.

Parameters

<i>runner</i>	NetworkRunner that despawned the NetworkBehaviour .
<i>hasState</i>	Whether the NetworkBehaviour has state.

Implemented in [HitboxRoot](#), and [NetworkBehaviour](#).

6.86 IElementReaderWriter< T > Interface Template Reference

Defines the interface for reading and writing elements in a byte array.

Public Member Functions

- int [GetElementHashCode](#) (T element)
Calculate the hash code of an element.
- int [GetElementWordCount](#) ()
Gets the word count of an element.
- T [Read](#) (byte *data, int index)
Reads an element from the specified index in the byte array.
- ref T [ReadRef](#) (byte *data, int index)
Reads a reference to an element from the specified index in the byte array.
- void [Write](#) (byte *data, int index, T element)
Writes an element to the specified index in the byte array.

6.86.1 Detailed Description

Defines the interface for reading and writing elements in a byte array.

Template Parameters

<i>T</i>	The type of the elements.
----------	---------------------------

6.86.2 Member Function Documentation

6.86.2.1 GetElementHashCode()

```
int GetElementHashCode (  
    T element )
```

Calculate the hash code of an element.

Parameters

<i>element</i>	
----------------	--

Returns

6.86.2.2 GetElementWordCount()

```
int GetElementWordCount ( )
```

Gets the word count of an element.

Returns

The word count of an element.

6.86.2.3 Read()

```
T Read (
    byte * data,
    int index )
```

Reads an element from the specified index in the byte array.

Parameters

<i>data</i>	The byte array.
<i>index</i>	The index of the element.

Returns

The element at the specified index.

6.86.2.4 ReadRef()

```
ref T ReadRef (
    byte * data,
    int index )
```

Reads a reference to an element from the specified index in the byte array.

Parameters

<i>data</i>	The byte array.
<i>index</i>	The index of the element.

Returns

A reference to the element at the specified index.

6.86.2.5 Write()

```
void Write (
    byte * data,
    int index,
    T element )
```

Writes an element to the specified index in the byte array.

Parameters

<i>data</i>	The byte array.
<i>index</i>	The index at which to write the element.
<i>element</i>	The element to write.

6.87 IFixedStorage Interface Reference

Interface for fixed storage types.

Inherited by [_128](#), [_16](#), [_2](#), [_256](#), [_32](#), [_4](#), [_512](#), [_64](#), and [_8](#).

6.87.1 Detailed Description

Interface for fixed storage types.

6.88 InputAuthorityGained Interface Reference

Interface for handling the event when the input authority is gained.

Inherits [IPublicFacingInterface](#).

Public Member Functions

- void [InputAuthorityGained](#) ()
Method to be called when the input authority is gained.

6.88.1 Detailed Description

Interface for handling the event when the input authority is gained.

6.88.2 Member Function Documentation

6.88.2.1 InputAuthorityGained()

```
void InputAuthorityGained ( )
```

Method to be called when the input authority is gained.

6.89 InputAuthorityLost Interface Reference

Interface for handling the event when the input authority is lost.

Inherits [IPublicFacingInterface](#).

Public Member Functions

- void [InputAuthorityLost](#) ()
Method to be called when the input authority is lost.

6.89.1 Detailed Description

Interface for handling the event when the input authority is lost.

6.89.2 Member Function Documentation

6.89.2.1 InputAuthorityLost()

```
void InputAuthorityLost ( )
```

Method to be called when the input authority is lost.

6.90 InterestEnter Interface Reference

Interface for handling the event when a player enters the area of interest.

Inherits [IPublicFacingInterface](#).

Public Member Functions

- void [InterestEnter](#) ([PlayerRef](#) player)
Method to be called when a player enters the area of interest.

6.90.1 Detailed Description

Interface for handling the event when a player enters the area of interest.

6.90.2 Member Function Documentation

6.90.2.1 InterestEnter()

```
void InterestEnter (  
    PlayerRef player )
```

Method to be called when a player enters the area of interest.

Parameters

<i>player</i>	The player who entered the area of interest.
---------------	--

6.91 InterestExit Interface Reference

Interface for handling the event when a player exits the area of interest.

Inherits [IPublicFacingInterface](#).

Public Member Functions

- void [InterestExit](#) ([PlayerRef](#) player)
Method to be called when a player exits the area of interest.

6.91.1 Detailed Description

Interface for handling the event when a player exits the area of interest.

6.91.2 Member Function Documentation

6.91.2.1 InterestExit()

```
void InterestExit (
    PlayerRef player )
```

Method to be called when a player exits the area of interest.

Parameters

<i>player</i>	The player who exited the area of interest.
---------------	---

6.92 ILocalPrefabCreated Interface Reference

Interface for handling the event when a local prefab is created.

Inherits [IPublicFacingInterface](#).

Public Member Functions

- void [LocalPrefabCreated](#) ()
Method to be called when a local prefab is created.

6.92.1 Detailed Description

Interface for handling the event when a local prefab is created.

6.92.2 Member Function Documentation

6.92.2.1 LocalPrefabCreated()

```
void LocalPrefabCreated ( )
```

Method to be called when a local prefab is created.

6.93 INetworkArray Interface Reference

Defines the interface for a networked array.

Inherits [IEnumerable](#).

Inherited by [NetworkArray< T >](#).

Properties

- object [this\[int index\]](#) [get, set]
Gets or sets the element at the specified index.

6.93.1 Detailed Description

Defines the interface for a networked array.

6.93.2 Property Documentation

6.93.2.1 this[int index]

```
object this[int index] [get], [set]
```

Gets or sets the element at the specified index.

Parameters

<i>index</i>	The zero-based index of the element to get or set.
--------------	--

6.94 INetworkAssetSource< T > Interface Template Reference

Interface for a network asset source.

Public Member Functions

- void [Acquire](#) (bool synchronous)
Acquires the network asset.
- void [Release](#) ()
Releases the network asset.
- T [WaitForResult](#) ()
Waits for the result of the network asset acquisition.

Properties

- string [Description](#) [get]
Gets the description of the network asset.
- bool [IsCompleted](#) [get]
Checks if the network asset acquisition is completed.

6.94.1 Detailed Description

Interface for a network asset source.

Template Parameters

<i>T</i>	Type of the network asset.
----------	----------------------------

6.94.2 Member Function Documentation

6.94.2.1 Acquire()

```
void Acquire (
    bool synchronous )
```

Acquires the network asset.

Parameters

<i>synchronous</i>	If true, the acquisition is done synchronously.
--------------------	---

6.94.2.2 Release()

```
void Release ( )
```

Releases the network asset.

6.94.2.3 WaitForResult()

```
T WaitForResult ( )
```

Waits for the result of the network asset acquisition.

Returns

The network asset.

6.94.3 Property Documentation

6.94.3.1 Description

```
string Description [get]
```

Gets the description of the network asset.

6.94.3.2 IsCompleted

```
bool IsCompleted [get]
```

Checks if the network asset acquisition is completed.

6.95 INetworkDictionary Interface Reference

Defines the interface for a networked dictionary.

Inherits IEnumerable.

Inherited by [NetworkDictionary< K, V >](#).

Public Member Functions

- void [Add](#) (object item)
Adds an item to the networked dictionary.

6.95.1 Detailed Description

Defines the interface for a networked dictionary.

6.95.2 Member Function Documentation

6.95.2.1 Add()

```
void Add (  
    object item )
```

Adds an item to the networked dictionary.

Parameters

<i>item</i>	The item to add to the dictionary.
-------------	------------------------------------

Implemented in [NetworkDictionary< K, V >](#).

6.96 INetworkInput Interface Reference

Flag interface for custom [NetworkInput](#) structs.

6.96.1 Detailed Description

Flag interface for custom [NetworkInput](#) structs.

6.97 INetworkLinkedList Interface Reference

Defines the interface for a networked linked list.

Inherits [IEnumerable](#).

Inherited by [NetworkLinkedList< T >](#).

Public Member Functions

- void [Add](#) (object item)
Adds an item to the networked linked list.

6.97.1 Detailed Description

Defines the interface for a networked linked list.

6.97.2 Member Function Documentation

6.97.2.1 Add()

```
void Add (
    object item )
```

Adds an item to the networked linked list.

Parameters

<i>item</i>	The item to add to the linked list.
-------------	-------------------------------------

Implemented in [NetworkLinkedList< T >](#).

6.98 INetworkObjectInitializer Interface Reference

Interface for initializing network objects.

Inherited by [NetworkObjectInitializerUnity](#).

Public Member Functions

- void [InitializeNetworkState](#) ([NetworkObject](#) networkObject)
Initializes the network object.

6.98.1 Detailed Description

Interface for initializing network objects.

6.98.2 Member Function Documentation

6.98.2.1 InitializeNetworkState()

```
void InitializeNetworkState (  
    NetworkObject networkObject )
```

Initializes the network object.

Parameters

<i>networkObject</i>	The network object to initialize.
----------------------	-----------------------------------

Implemented in [NetworkObjectInitializerUnity](#).

6.99 INetworkObjectProvider Interface Reference

Interface which defines the handlers for [NetworkRunner](#) [Spawn\(\)](#) and [Despawn\(\)](#) actions. Passing an instance of this interface to [NetworkRunner.StartGame\(StartGameArgs\)](#) as the [StartGameArgs.ObjectProvider](#) argument value will assign that instance as the handler for runner [Spawn\(\)](#) and [Despawn\(\)](#) actions. By default (if [StartGameArgs.ObjectProvider](#) == null) actions will use [Instantiate\(\)](#), and [Despawn\(\)](#) actions will use [Destroy\(\)](#).

Inherited by [NetworkObjectProviderDummy](#).

Public Member Functions

- [NetworkObjectAcquireResult AcquirePrefabInstance](#) ([NetworkRunner](#) runner, in [NetworkPrefabAcquireContext](#) context, out [NetworkObject](#) result)

Acquires an instance of a prefab for a network object.
- [NetworkPrefabId GetPrefabId](#) ([NetworkRunner](#) runner, [NetworkObjectGuid](#) prefabGuid)

Translates guid into prefab id.
- void [Initialize](#) ([NetworkRunner](#) networkRunner)
- void [ReleaseInstance](#) ([NetworkRunner](#) runner, in [NetworkObjectReleaseContext](#) context)

Releases an instance of a network object.
- void [Shutdown](#) ([NetworkRunner](#) networkRunner)

6.99.1 Detailed Description

Interface which defines the handlers for [NetworkRunner](#) [Spawn\(\)](#) and [Despawn\(\)](#) actions. Passing an instance of this interface to [NetworkRunner.StartGame\(StartGameArgs\)](#) as the [StartGameArgs.ObjectProvider](#) argument value will assign that instance as the handler for runner [Spawn\(\)](#) and [Despawn\(\)](#) actions. By default (if [StartGameArgs.ObjectProvider](#) == null) actions will use [Instantiate\(\)](#), and [Despawn\(\)](#) actions will use [Destroy\(\)](#).

6.99.2 Member Function Documentation

6.99.2.1 AcquirePrefabInstance()

```
NetworkObjectAcquireResult AcquirePrefabInstance (
    NetworkRunner runner,
    in NetworkPrefabAcquireContext context,
    out NetworkObject result )
```

Acquires an instance of a prefab for a network object.

Parameters

<i>runner</i>	The NetworkRunner that manages the network objects.
<i>context</i>	The context that provides information for acquiring the prefab instance.
<i>result</i>	The acquired NetworkObject instance.

Returns

A [NetworkObjectAcquireResult](#) indicating the result of the operation.

Implemented in [NetworkObjectProviderDummy](#).

6.99.2.2 GetPrefabId()

```
NetworkPrefabId GetPrefabId (
    NetworkRunner runner,
    NetworkObjectGuid prefabGuid )
```

Translates guid into prefab id.

Parameters

<i>runner</i>	
<i>prefabGuid</i>	

Returns

Implemented in [NetworkObjectProviderDummy](#).

6.99.2.3 Initialize()

```
void Initialize (
    NetworkRunner networkRunner )
```

Parameters

<i>networkRunner</i>	
----------------------	--

6.99.2.4 ReleaseInstance()

```
void ReleaseInstance (
    NetworkRunner runner,
    in NetworkObjectReleaseContext context )
```

Releases an instance of a network object.

Parameters

<i>runner</i>	The NetworkRunner that manages the network objects.
<i>context</i>	The context that provides information for releasing the network object instance.

Implemented in [NetworkObjectProviderDummy](#).

6.99.2.5 Shutdown()

```
void Shutdown (
    NetworkRunner networkRunner )
```

Parameters

<code>networkRunner</code>	
----------------------------	--

6.100 INetworkPrefabSource Interface Reference

Interface for a network prefab source.

Inherits [INetworkAssetSource](#)< [NetworkObject](#) >.

Properties

- [NetworkObjectGuid AssetGuid](#) [get]
Gets the GUID of the network object asset.

Additional Inherited Members

6.100.1 Detailed Description

Interface for a network prefab source.

6.100.2 Property Documentation

6.100.2.1 AssetGuid

```
NetworkObjectGuid AssetGuid [get]
```

Gets the GUID of the network object asset.

6.101 INetworkRunnerCallbacks Interface Reference

Interface for [NetworkRunner](#) callbacks. Register a class/struct instance which implements this interface with `NetworkRunner.AddCallbacks(INetworkRunnerCallbacks[])`.

Inherits [IPublicFacingInterface](#).

Inherited by [NetworkDelegates](#), and [NetworkEvents](#).

Public Member Functions

- void [OnConnectedToServer](#) ([NetworkRunner](#) runner)
 - Callback when [NetworkRunner](#) successfully connects to a server or host.*
- void [OnConnectFailed](#) ([NetworkRunner](#) runner, [NetAddress](#) remoteAddress, [NetConnectFailedReason](#) reason)
 - Callback when [NetworkRunner](#) fails to connect to a server or host.*
- void [OnConnectRequest](#) ([NetworkRunner](#) runner, [NetworkRunnerCallbackArgs.ConnectRequest](#) request, byte[] token)
 - Callback when [NetworkRunner](#) receives a Connection Request from a Remote Client*
- void [OnCustomAuthenticationResponse](#) ([NetworkRunner](#) runner, Dictionary< string, object > data)
 - Callback is invoked when the Authentication procedure returns a response from the Authentication Server*
- void [OnDisconnectedFromServer](#) ([NetworkRunner](#) runner, [NetDisconnectReason](#) reason)
 - Callback when [NetworkRunner](#) disconnects from a server or host.*
- void [OnHostMigration](#) ([NetworkRunner](#) runner, [HostMigrationToken](#) hostMigrationToken)
 - Callback is invoked when the Host Migration process has started*
- void [OnInput](#) ([NetworkRunner](#) runner, [NetworkInput](#) input)
 - Callback from [NetworkRunner](#) that polls for user inputs. The [NetworkInput](#) that is supplied expects:*
- void [OnInputMissing](#) ([NetworkRunner](#) runner, [PlayerRef](#) player, [NetworkInput](#) input)
 - Callback from [NetworkRunner](#) when an input is missing.*
- void [OnObjectEnterAOI](#) ([NetworkRunner](#) runner, [NetworkObject](#) obj, [PlayerRef](#) player)
 - Callback from a [NetworkRunner](#) when a new [NetworkObject](#) has entered the Area of Interest*
- void [OnObjectExitAOI](#) ([NetworkRunner](#) runner, [NetworkObject](#) obj, [PlayerRef](#) player)
 - Callback from a [NetworkRunner](#) when a new [NetworkObject](#) has exit the Area of Interest*
- void [OnPlayerJoined](#) ([NetworkRunner](#) runner, [PlayerRef](#) player)
 - Callback from a [NetworkRunner](#) when a new player has joined.*
- void [OnPlayerLeft](#) ([NetworkRunner](#) runner, [PlayerRef](#) player)
 - Callback from a [NetworkRunner](#) when a player has disconnected.*
- void [OnReliableDataProgress](#) ([NetworkRunner](#) runner, [PlayerRef](#) player, [ReliableKey](#) key, float progress)
 - Callback is invoked when a Reliable Data Stream is being received, reporting its progress*
- void [OnReliableDataReceived](#) ([NetworkRunner](#) runner, [PlayerRef](#) player, [ReliableKey](#) key, ArraySegment< byte > data)
 - Callback is invoked when a Reliable Data Stream has been received*
- void [OnSceneLoadDone](#) ([NetworkRunner](#) runner)
 - Callback is invoked when a Scene Load has finished*
- void [OnSceneLoadStart](#) ([NetworkRunner](#) runner)
 - Callback is invoked when a Scene Load has started*
- void [OnSessionListUpdated](#) ([NetworkRunner](#) runner, List< [SessionInfo](#) > sessionList)
 - This callback is invoked when a new List of Sessions is received from Photon Cloud*
- void [OnShutdown](#) ([NetworkRunner](#) runner, [ShutdownReason](#) shutdownReason)
 - Called when the runner is shutdown*
- void [OnUserSimulationMessage](#) ([NetworkRunner](#) runner, [SimulationMessagePtr](#) message)
 - This callback is invoked when a manually dispatched simulation message is received from a remote peer*

6.101.1 Detailed Description

Interface for [NetworkRunner](#) callbacks. Register a class/struct instance which implements this interface with [NetworkRunner.AddCallbacks\(INetworkRunnerCallbacks\[\]\)](#).

6.101.2 Member Function Documentation

6.101.2.1 OnConnectedToServer()

```
void OnConnectedToServer (
    NetworkRunner runner )
```

Callback when [NetworkRunner](#) successfully connects to a server or host.

6.101.2.2 OnConnectFailed()

```
void OnConnectFailed (
    NetworkRunner runner,
    NetAddress remoteAddress,
    NetConnectFailedReason reason )
```

Callback when [NetworkRunner](#) fails to connect to a server or host.

6.101.2.3 OnConnectRequest()

```
void OnConnectRequest (
    NetworkRunner runner,
    NetworkRunnerCallbackArgs.ConnectRequest request,
    byte[] token )
```

Callback when [NetworkRunner](#) receives a Connection Request from a Remote Client

Parameters

<i>runner</i>	Local NetworkRunner
<i>request</i>	Request information
<i>token</i>	Request Token

6.101.2.4 OnCustomAuthenticationResponse()

```
void OnCustomAuthenticationResponse (
    NetworkRunner runner,
    Dictionary< string, object > data )
```

Callback is invoked when the Authentication procedure returns a response from the Authentication Server

Parameters

<i>runner</i>	The runner this object exists on
<i>data</i>	Custom Authentication Reply Values

6.101.2.5 OnDisconnectedFromServer()

```
void OnDisconnectedFromServer (
    NetworkRunner runner,
    NetDisconnectReason reason )
```

Callback when [NetworkRunner](#) disconnects from a server or host.

6.101.2.6 OnHostMigration()

```
void OnHostMigration (
    NetworkRunner runner,
    HostMigrationToken hostMigrationToken )
```

Callback is invoked when the Host Migration process has started

Parameters

<i>runner</i>	The runner this object exists on
<i>hostMigrationToken</i>	Migration Token that stores all necessary information to restart the Fusion Runner

6.101.2.7 OnInput()

```
void OnInput (
    NetworkRunner runner,
    NetworkInput input )
```

Callback from [NetworkRunner](#) that polls for user inputs. The [NetworkInput](#) that is supplied expects:

```
input.Set(new CustomINetworkInput() { /* your values */ });
```

6.101.2.8 OnInputMissing()

```
void OnInputMissing (
    NetworkRunner runner,
    PlayerRef player,
    NetworkInput input )
```

Callback from [NetworkRunner](#) when an input is missing.

Parameters

<i>runner</i>	NetworkRunner reference
<i>player</i>	PlayerRef reference which the input is missing from
<i>input</i>	NetworkInput reference which is missing

6.101.2.9 OnObjectEnterAOI()

```
void OnObjectEnterAOI (
    NetworkRunner runner,
    NetworkObject obj,
    PlayerRef player )
```

Callback from a [NetworkRunner](#) when a new [NetworkObject](#) has entered the Area of Interest

Parameters

<i>runner</i>	NetworkRunner reference
<i>obj</i>	NetworkObject reference
<i>player</i>	PlayerRef reference

6.101.2.10 OnObjectExitAOI()

```
void OnObjectExitAOI (
    NetworkRunner runner,
    NetworkObject obj,
    PlayerRef player )
```

Callback from a [NetworkRunner](#) when a new [NetworkObject](#) has exit the Area of Interest

Parameters

<i>runner</i>	NetworkRunner reference
<i>obj</i>	NetworkObject reference
<i>player</i>	PlayerRef reference

6.101.2.11 OnPlayerJoined()

```
void OnPlayerJoined (
    NetworkRunner runner,
    PlayerRef player )
```

Callback from a [NetworkRunner](#) when a new player has joined.

6.101.2.12 OnPlayerLeft()

```
void OnPlayerLeft (
    NetworkRunner runner,
    PlayerRef player )
```

Callback from a [NetworkRunner](#) when a player has disconnected.

6.101.2.13 OnReliableDataProgress()

```
void OnReliableDataProgress (
    NetworkRunner runner,
    PlayerRef player,
    ReliableKey key,
    float progress )
```

Callback is invoked when a Reliable Data Stream is being received, reporting its progress

Parameters

<i>runner</i>	NetworkRunner reference
<i>player</i>	Which PlayerRef the stream is being sent from
<i>key</i>	ReliableKey reference that identifies the data stream
<i>progress</i>	Progress of the stream

6.101.2.14 OnReliableDataReceived()

```
void OnReliableDataReceived (
    NetworkRunner runner,
    PlayerRef player,
    ReliableKey key,
    ArraySegment< byte > data )
```

Callback is invoked when a Reliable Data Stream has been received

Parameters

<i>runner</i>	NetworkRunner reference
<i>player</i>	Which PlayerRef the stream was sent from
<i>key</i>	ReliableKey reference that identifies the data stream
<i>data</i>	Data received

6.101.2.15 OnSceneLoadDone()

```
void OnSceneLoadDone (
    NetworkRunner runner )
```

Callback is invoked when a Scene Load has finished

Parameters

<i>runner</i>	NetworkRunner reference
---------------	-------------------------

6.101.2.16 OnSceneLoadStart()

```
void OnSceneLoadStart (
    NetworkRunner runner )
```

Callback is invoked when a Scene Load has started

Parameters

<i>runner</i>	NetworkRunner reference
---------------	-------------------------

6.101.2.17 OnSessionListUpdated()

```
void OnSessionListUpdated (
    NetworkRunner runner,
    List< SessionInfo > sessionList )
```

This callback is invoked when a new List of Sessions is received from Photon Cloud

Parameters

<i>runner</i>	The runner this object exists on
<i>sessionList</i>	Updated list of Session

6.101.2.18 OnShutdown()

```
void OnShutdown (
    NetworkRunner runner,
    ShutdownReason shutdownReason )
```

Called when the runner is shutdown

Parameters

<i>runner</i>	The runner being shutdown
<i>shutdownReason</i>	Describes the reason Fusion was Shutdown

6.101.2.19 OnUserSimulationMessage()

```
void OnUserSimulationMessage (
    NetworkRunner runner,
    SimulationMessagePtr message )
```

This callback is invoked when a manually dispatched simulation message is received from a remote peer

Parameters

<i>runner</i>	The runner this message is for
<i>message</i>	The message pointer

6.102 INetworkRunnerUpdater Interface Reference

Interface which defines the handlers for [NetworkRunner](#) Updates. An implementation is responsible for calling [NetworkRunner.UpdateInternal\(double\)](#) and [NetworkRunner.RenderInternal](#) periodically.

Inherited by [NetworkRunnerUpdaterDefault](#), and [NetworkRunnerUpdaterDummy](#).

Public Member Functions

- void [Initialize](#) ([NetworkRunner](#) runner)
Called when the [NetworkRunner](#) is started.
- void [Shutdown](#) ([NetworkRunner](#) runner)
Called when the [NetworkRunner](#) is stopped.

6.102.1 Detailed Description

Interface which defines the handlers for [NetworkRunner](#) Updates. An implementation is responsible for calling [NetworkRunner.UpdateInternal\(double\)](#) and [NetworkRunner.RenderInternal](#) periodically.

An instance of this interface can be passed to [NetworkRunner.StartGame\(StartGameArgs\)](#) as the [StartGameArgs.Updater](#). By default (if [StartGameArgs.Updater](#) == null) [Fusion](#) will use [NetworkRunnerUpdaterDefault](#), which invokes [NetworkRunner.UpdateInternal\(double\)](#) before script's Update and [NetworkRunner.RenderInternal](#) before Late↔ Update.

6.102.2 Member Function Documentation

6.102.2.1 Initialize()

```
void Initialize (
    NetworkRunner runner )
```

Called when the [NetworkRunner](#) is started.

Parameters

<code>runner</code>	The NetworkRunner instance.
---------------------	---

6.102.2.2 Shutdown()

```
void Shutdown (
    NetworkRunner runner )
```

Called when the [NetworkRunner](#) is stopped.

Parameters

<code>runner</code>	The NetworkRunner instance.
---------------------	---

6.103 INetworkSceneManager Interface Reference

Interface for a [NetworkRunner](#) scene manager. A scene manager is responsible for loading and unloading scenes

Inherited by [NetworkSceneManagerDummy](#).

Public Member Functions

- [SceneRef GetSceneRef](#) (GameObject gameObject)
Gets a [SceneRef](#) for the scene that the given [GameObject](#) belongs to.
- [SceneRef GetSceneRef](#) (string sceneNameOrPath)
Gets a [SceneRef](#) for the given scene name or path.
- void [Initialize](#) ([NetworkRunner](#) runner)
Callback for initialization
- bool [IsRunnerScene](#) (Scene scene)
Signals if the given scene is the main runner scene. Mostly used for Multipeer logic
- [NetworkSceneAsyncOp LoadScene](#) ([SceneRef](#) sceneRef, [NetworkLoadSceneParameters](#) parameters)
Loads a given scene with the specified parameters.
- void [MakeDontDestroyOnLoad](#) (GameObject obj)
*Mark an object as *DontDestroyOnLoad*.*
- bool [MoveGameObjectToScene](#) (GameObject gameObject, [SceneRef](#) sceneRef)
Move a [GameObject](#) to a desired scene.
- bool [OnSceneInfoChanged](#) ([NetworkSceneInfo](#) sceneInfo, [NetworkSceneInfoChangeSource](#) changeSource)

Implement this method and return true if you want to handle scene info changes manually. Return false if the default scene info change handling should be done by the [NetworkRunner](#) instead.

- void [Shutdown](#) ()
Callback for shutdown and clean up
- bool [TryGetPhysicsScene2D](#) (out PhysicsScene2D scene2D)
Tries to get the physics scene 2D.
- bool [TryGetPhysicsScene3D](#) (out PhysicsScene scene3D)
Tries to get the physics scene 3D.
- [NetworkSceneAsyncOp UnloadScene](#) ([SceneRef](#) sceneRef)
Unloads a given scene.

Properties

- bool [IsBusy](#) [get]
Signals if the [INetworkSceneManager](#) instance is busy with any scene loading operations
- Scene [MainRunnerScene](#) [get]
The main scene of the [NetworkRunner](#). Mostly used for Multipeer logic

6.103.1 Detailed Description

Interface for a [NetworkRunner](#) scene manager. A scene manager is responsible for loading and unloading scenes

6.103.2 Member Function Documentation

6.103.2.1 GetSceneRef() [1/2]

```
SceneRef GetSceneRef (
    GameObject gameObject )
```

Gets a [SceneRef](#) for the scene that the given [GameObject](#) belongs to.

6.103.2.2 GetSceneRef() [2/2]

```
SceneRef GetSceneRef (
    string sceneNameOrPath )
```

Gets a [SceneRef](#) for the given scene name or path.

6.103.2.3 Initialize()

```
void Initialize (
    NetworkRunner runner )
```

Callback for initialization

6.103.2.4 IsRunnerScene()

```
bool IsRunnerScene (
    Scene scene )
```

Signals if the given scene is the main runner scene. Mostly used for Multipeer logic

6.103.2.5 LoadScene()

```
NetworkSceneAsyncOp LoadScene (
    SceneRef sceneRef,
    NetworkLoadSceneParameters parameters )
```

Loads a given scene with the specified parameters.

Returns

Returns a [NetworkSceneAsyncOp](#) that can be waited

6.103.2.6 MakeDontDestroyOnLoad()

```
void MakeDontDestroyOnLoad (
    GameObject obj )
```

Mark an object as DontDestroyOnLoad.

6.103.2.7 MoveGameObjectToScene()

```
bool MoveGameObjectToScene (
    GameObject gameObject,
    SceneRef sceneRef )
```

Move a GameObject to a desired scene.

Returns

Return true if the operation was successfully

6.103.2.8 OnSceneInfoChanged()

```
bool OnSceneInfoChanged (
    NetworkSceneInfo sceneInfo,
    NetworkSceneInfoChangeSource changeSource )
```

Implement this method and return true if you want to handle scene info changes manually. Return false if the default scene info change handling should be done by the [NetworkRunner](#) instead.

Returns

Return true if a custom handling is provided, false otherwise to use the default one

6.103.2.9 Shutdown()

```
void Shutdown ( )
```

Callback for shutdown and clean up

6.103.2.10 TryGetPhysicsScene2D()

```
bool TryGetPhysicsScene2D (
    out PhysicsScene2D scene2D )
```

Tries to get the physics scene 2D.

Returns

Returns true if the operation was successfully

6.103.2.11 TryGetPhysicsScene3D()

```
bool TryGetPhysicsScene3D (
    out PhysicsScene scene3D )
```

Tries to get the physics scene 3D.

Returns

Returns true if the operation was successfully

6.103.2.12 UnloadScene()

```
NetworkSceneAsyncOp UnloadScene (
    SceneRef sceneRef )
```

Unloads a given scene.

Returns

Returns a [NetworkSceneAsyncOp](#) that can be waited

6.103.3 Property Documentation

6.103.3.1 IsBusy

```
bool IsBusy [get]
```

Signals if the [INetworkSceneManager](#) instance is busy with any scene loading operations

6.103.3.2 MainRunnerScene

```
Scene MainRunnerScene [get]
```

The main scene of the [NetworkRunner](#). Mostly used for Multipeer logic

6.104 INetworkStruct Interface Reference

Base interface for all [Fusion](#) Network Structs

Inherited by [Angle](#), [BitSet128](#), [BitSet192](#), [BitSet256](#), [BitSet512](#), [BitSet64](#), [FloatCompressed](#), [NetworkBehaviourId](#), [NetworkBool](#), [NetworkButtons](#), [NetworkId](#), [NetworkObjectGuid](#), [NetworkObjectHeader](#), [NetworkObjectNestingKey](#), [NetworkObjectTypeId](#), [NetworkPhysicsInfo](#), [NetworkPrefabId](#), [NetworkPrefabRef](#), [NetworkRNG](#), [NetworkSceneInfo](#), [NetworkString< TSize >](#), [NetworkTRSPData](#), [PlayerRef](#), [Ptr](#), [QuaternionCompressed](#), [SceneRef](#), [TickTimer](#), [Vector2Compressed](#), [Vector3Compressed](#), [Vector4Compressed](#), [_128](#), [_16](#), [_2](#), [_256](#), [_32](#), [_4](#), [_512](#), [_64](#), and [_8](#).

6.104.1 Detailed Description

Base interface for all [Fusion](#) Network Structs

6.105 INetworkTRSPTeleport Interface Reference

Implement this interface on a [NetworkTRSP](#) implementation to indicate it can be teleported.

Inherited by [NetworkTransform](#).

Public Member Functions

- void [Teleport](#) (Vector3? position=null, Quaternion? rotation=null)
Teleports to the indicated values, and network the Teleport event.

6.105.1 Detailed Description

Implement this interface on a [NetworkTRSP](#) implementation to indicate it can be teleported.

6.105.2 Member Function Documentation

6.105.2.1 Teleport()

```
void Teleport (
    Vector3? position = null,
    Quaternion? rotation = null )
```

Teleports to the indicated values, and network the Teleport event.

Implemented in [NetworkTransform](#).

6.106 InlineHelpAttribute Class Reference

If applied to a field, checks if there is a help text for the field or the field's type and shows it in the inspector.

Inherits [DecoratingPropertyAttribute](#).

Public Member Functions

- [InlineHelpAttribute](#) ()
Initializes a new instance of the [InlineHelpAttribute](#) class.

Properties

- bool [ShowTypeHelp](#) = true [get, set]
Gets or sets a value indicating whether to show help for the type.

Additional Inherited Members

6.106.1 Detailed Description

If applied to a field, checks if there is a help text for the field or the field's type and shows it in the inspector.

6.106.2 Constructor & Destructor Documentation

6.106.2.1 InlineHelpAttribute()

```
InlineHelpAttribute ( )
```

Initializes a new instance of the [InlineHelpAttribute](#) class.

6.106.3 Property Documentation

6.106.3.1 ShowTypeHelp

```
bool ShowTypeHelp = true [get], [set]
```

Gets or sets a value indicating whether to show help for the type.

6.107 IUnitySurrogate Interface Reference

Represents an interface for Unity surrogates. This interface provides methods for reading and writing data.

Inherited by [IUnityValueSurrogate< T >](#), and [UnitySurrogateBase](#).

Public Member Functions

- void [Read](#) (int *data, int capacity)
Reads data from a specified memory location.
- void [Write](#) (int *data, int capacity)
Writes data to a specified memory location.

6.107.1 Detailed Description

Represents an interface for Unity surrogates. This interface provides methods for reading and writing data.

6.107.2 Member Function Documentation

6.107.2.1 Read()

```
void Read (  
    int * data,  
    int capacity )
```

Reads data from a specified memory location.

Parameters

<i>data</i>	The memory location to read from.
<i>capacity</i>	The number of elements to read.

Implemented in [UnityValueSurrogate< T, TReaderWriter >](#), [UnitySurrogateBase](#), [UnityLinkedListSurrogate< T, ReaderWriter >](#), [UnityDictionarySurrogate< TKeyType, TKeyReaderWriter, TValueType, TValueReaderWriter >](#), and [UnityArraySurrogate< T, ReaderWriter >](#).

6.107.2.2 Write()

```
void Write (
    int * data,
    int capacity )
```

Writes data to a specified memory location.

Parameters

<i>data</i>	The memory location to write to.
<i>capacity</i>	The number of elements to write.

Implemented in [UnityValueSurrogate< T, TReaderWriter >](#), [UnitySurrogateBase](#), [UnityLinkedListSurrogate< T, ReaderWriter >](#), [UnityDictionarySurrogate< TKeyType, TKeyReaderWriter, TValueType, TValueReaderWriter >](#), and [UnityArraySurrogate< T, ReaderWriter >](#).

6.108 IUnityValueSurrogate< T > Interface Template Reference

Represents an interface for Unity value surrogates. This interface provides a property for accessing and modifying the data.

Inherits [IUnitySurrogate](#).

Inherited by [UnityValueSurrogate< T, TReaderWriter >](#).

Properties

- [T DataProperty](#) [get, set]
Gets or sets the data for the Unity value surrogate.

Additional Inherited Members**6.108.1 Detailed Description**

Represents an interface for Unity value surrogates. This interface provides a property for accessing and modifying the data.

Template Parameters

<i>T</i>	The type of the data.
----------	-----------------------

6.108.2 Property Documentation

6.108.2.1 DataProperty

`T DataProperty [get], [set]`

Gets or sets the data for the Unity value surrogate.

6.109 UnityArraySurrogate< T, ReaderWriter > Class Template Reference

A base class for Unity array surrogates.

Inherits [UnitySurrogateBase](#).

Public Member Functions

- override void [Init](#) (int capacity)
Initializes the [UnityArraySurrogate](#) with a specified capacity.
- override void [Read](#) (int *data, int capacity)
Reads data into the [UnityArraySurrogate](#) from a specified memory location.
- override void [Write](#) (int *data, int capacity)
Writes data from the [UnityArraySurrogate](#) to a specified memory location.

Properties

- abstract `T[] DataProperty [get, set]`
Gets or sets the data array for the [UnityArraySurrogate](#).

6.109.1 Detailed Description

A base class for Unity array surrogates.

Template Parameters

<i>T</i>	Unmanaged type of the array elements.
<i>ReaderWriter</i>	Unmanaged type of the reader/writer for the array elements.

Type Constraints

T : *unmanaged*

ReaderWriter : *unmanaged*

ReaderWriter : [IElementReaderWriter](#)<*T*>

6.109.2 Member Function Documentation

6.109.2.1 Init()

```
override void Init (
    int capacity ) [virtual]
```

Initializes the [UnityArraySurrogate](#) with a specified capacity.

Parameters

<i>capacity</i>	The capacity to initialize the UnityArraySurrogate with.
-----------------	--

Implements [UnitySurrogateBase](#).

6.109.2.2 Read()

```
override void Read (
    int * data,
    int capacity ) [virtual]
```

Reads data into the [UnityArraySurrogate](#) from a specified memory location.

Parameters

<i>data</i>	The memory location to read from.
<i>capacity</i>	The number of elements to read.

Implements [UnitySurrogateBase](#).

6.109.2.3 Write()

```
override void Write (
    int * data,
    int capacity ) [virtual]
```

Writes data from the [UnityArraySurrogate](#) to a specified memory location.

Parameters

<i>data</i>	The memory location to write to.
<i>capacity</i>	The number of elements to write.

Implements [UnitySurrogateBase](#).

6.109.3 Property Documentation

6.109.3.1 DataProperty

```
abstract T [] DataProperty [get], [set]
```

Gets or sets the data array for the [UnityArraySurrogate](#).

6.110 UnityDictionarySurrogate< TKeyType, TKeyReaderWriter, TValueType, TValueReaderWriter > Class Template Reference

A surrogate for serializing a dictionary.

Inherits [UnitySurrogateBase](#).

Public Member Functions

- override void [Init](#) (int capacity)
Initializes the [UnityDictionarySurrogate](#) with a specified capacity.
- override void [Read](#) (int *data, int capacity)
Reads data into the [UnityDictionarySurrogate](#) from a specified memory location.
- override void [Write](#) (int *data, int capacity)
Writes data from the [UnityDictionarySurrogate](#) to a specified memory location.

Properties

- abstract [SerializableDictionary](#)< TKeyType, TValueType > [DataProperty](#) [get, set]
Gets or sets the data property.

6.110.1 Detailed Description

A surrogate for serializing a dictionary.

Template Parameters

<i>TKeyType</i>	The type of the key.
<i>TKeyReaderWriter</i>	The type of the key reader writer.
<i>TValueType</i>	The type of the value.
<i>TValueReaderWriter</i>	The type of the value reader writer.

See also

[Fusion.Internal.UnitySurrogateBase](#)

Type Constraints

***TKeyType* : unmanaged**

***TKeyReaderWriter* : unmanaged**

***TKeyReaderWriter* : [IElementReaderWriter](#)<*TKeyType*>**

***TValueType* : unmanaged**

***TValueReaderWriter* : unmanaged**

***TValueReaderWriter* : [IElementReaderWriter](#)<*TValueType*>**

6.110.2 Member Function Documentation

6.110.2.1 Init()

```
override void Init (
    int capacity ) [virtual]
```

Initializes the [UnityDictionarySurrogate](#) with a specified capacity.

Parameters

<i>capacity</i>	The capacity to initialize the UnityDictionarySurrogate with.
-----------------	---

Implements [UnitySurrogateBase](#).

6.110.2.2 Read()

```
override void Read (
    int * data,
    int capacity ) [virtual]
```

Reads data into the [UnityDictionarySurrogate](#) from a specified memory location.

Parameters

<i>data</i>	The memory location to read from.
<i>capacity</i>	The number of elements to read.

Implements [UnitySurrogateBase](#).

6.110.2.3 Write()

```
override void Write (  
    int * data,  
    int capacity ) [virtual]
```

Writes data from the [UnityDictionarySurrogate](#) to a specified memory location.

Parameters

<i>data</i>	The memory location to write to.
<i>capacity</i>	The number of elements to write.

Implements [UnitySurrogateBase](#).

6.110.3 Property Documentation

6.110.3.1 DataProperty

```
abstract SerializableDictionary<TKeyType, TValueType> DataProperty [get], [set]
```

Gets or sets the data property.

6.111 UnityLinkedListSurrogate< T, ReaderWriter > Class Template Reference

A surrogate for serializing a linked list.

Inherits [UnitySurrogateBase](#).

Public Member Functions

- override void [Init](#) (int capacity)
Initializes the [UnityLinkedListSurrogate](#) with a specified capacity.
- override void [Read](#) (int *data, int capacity)
Reads data into the [UnityLinkedListSurrogate](#) from a specified memory location.
- override void [Write](#) (int *data, int capacity)
Writes data from the [UnityLinkedListSurrogate](#) to a specified memory location.

Properties

- abstract T[] [DataProperty](#) [get, set]
Gets or sets the data property.

6.111.1 Detailed Description

A surrogate for serializing a linked list.

Template Parameters

<i>T</i>	The type of the elements in the linked list.
<i>ReaderWriter</i>	The type of the reader writer for the elements in the linked list.

Type Constraints

***T* : [unmanaged](#)**
***ReaderWriter* : [unmanaged](#)**
***ReaderWriter* : [IElementReaderWriter](#)<*T*>**

6.111.2 Member Function Documentation

6.111.2.1 Init()

```
override void Init (
    int capacity ) [virtual]
```

Initializes the [UnityLinkedListSurrogate](#) with a specified capacity.

Parameters

<i>capacity</i>	The capacity to initialize the UnityLinkedListSurrogate with.
-----------------	---

Implements [UnitySurrogateBase](#).

6.111.2.2 Read()

```
override void Read (
    int * data,
    int capacity ) [virtual]
```

Reads data into the [UnityLinkedListSurrogate](#) from a specified memory location.

Parameters

<i>data</i>	The memory location to read from.
<i>capacity</i>	The number of elements to read.

Implements [UnitySurrogateBase](#).

6.111.2.3 Write()

```
override void Write (
    int * data,
    int capacity ) [virtual]
```

Writes data from the [UnityLinkedListSurrogate](#) to a specified memory location.

Parameters

<i>data</i>	The memory location to write to.
<i>capacity</i>	The number of elements to write.

Implements [UnitySurrogateBase](#).

6.111.3 Property Documentation

6.111.3.1 DataProperty

```
abstract T [] DataProperty [get], [set]
```

Gets or sets the data property.

6.112 UnitySurrogateBase Class Reference

Represents a base class for Unity surrogates. This class is serializable and provides abstract methods for reading, writing, and initializing data.

Inherits [IUnitySurrogate](#).

Inherited by [UnityArraySurrogate< T, ReaderWriter >](#), [UnityDictionarySurrogate< TKeyType, TKeyReaderWriter, TValueType, TValueReaderWriter >](#), [UnityLinkedListSurrogate< T, ReaderWriter >](#), and [UnityValueSurrogate< T, TReaderWriter >](#).

Public Member Functions

- abstract void [Init](#) (int capacity)
Initializes the [UnitySurrogateBase](#) with a specified capacity.
- abstract void [Read](#) (int *data, int capacity)
Reads data from a specified memory location into the [UnitySurrogateBase](#).
- abstract void [Write](#) (int *data, int capacity)
Writes data from the [UnitySurrogateBase](#) to a specified memory location.

6.112.1 Detailed Description

Represents a base class for Unity surrogates. This class is serializable and provides abstract methods for reading, writing, and initializing data.

6.112.2 Member Function Documentation

6.112.2.1 Init()

```
abstract void Init (
    int capacity ) [pure virtual]
```

Initializes the [UnitySurrogateBase](#) with a specified capacity.

Parameters

<i>capacity</i>	The capacity to initialize the UnitySurrogateBase with.
-----------------	---

Implemented in [UnityValueSurrogate< T, TReaderWriter >](#), [UnityLinkedListSurrogate< T, ReaderWriter >](#), [UnityDictionarySurrogate< TKeyType, TKeyReaderWriter, TValueType, TValueReaderWriter >](#), and [UnityArraySurrogate< T, ReaderWriter >](#).

6.112.2.2 Read()

```
abstract void Read (
    int * data,
    int capacity ) [pure virtual]
```

Reads data from a specified memory location into the [UnitySurrogateBase](#).

Parameters

<i>data</i>	The memory location to read from.
<i>capacity</i>	The number of elements to read.

Implements [IUnitySurrogate](#).

Implemented in [UnityValueSurrogate< T, TReaderWriter >](#), [UnityLinkedListSurrogate< T, ReaderWriter >](#), [UnityDictionarySurrogate< TKeyType, TKeyReaderWriter, TValueType, TValueReaderWriter >](#), and [UnityArraySurrogate< T, ReaderWriter >](#).

6.112.2.3 Write()

```
abstract void Write (
    int * data,
    int capacity ) [pure virtual]
```

Writes data from the [UnitySurrogateBase](#) to a specified memory location.

Parameters

<i>data</i>	The memory location to write to.
<i>capacity</i>	The number of elements to write.

Implements [IUnitySurrogate](#).

Implemented in [UnityValueSurrogate< T, TReaderWriter >](#), [UnityLinkedListSurrogate< T, ReaderWriter >](#), [UnityDictionarySurrogate< TKeyType, TKeyReaderWriter, TValueType, TValueReaderWriter >](#), and [UnityArraySurrogate< T, ReaderWriter >](#).

6.113 UnityValueSurrogate< T, TReaderWriter > Class Template Reference

Represents a base class for Unity value surrogates. This class is serializable and provides methods for reading, writing, and initializing data.

Inherits [UnitySurrogateBase](#), and [IUnityValueSurrogate< T >](#).

Public Member Functions

- override void [Init](#) (int capacity)
Initializes the [UnityValueSurrogate](#) with a specified capacity.
- override void [Read](#) (int *data, int capacity)
Reads data into the [UnityValueSurrogate](#) from a specified memory location.
- override void [Write](#) (int *data, int capacity)
Writes data from the [UnityValueSurrogate](#) to a specified memory location.

Properties

- abstract T [DataProperty](#) [get, set]
Gets or sets the data for the [UnityValueSurrogate](#).

6.113.1 Detailed Description

Represents a base class for Unity value surrogates. This class is serializable and provides methods for reading, writing, and initializing data.

Template Parameters

<i>T</i>	The type of the data.
<i>TReaderWriter</i>	The type of the reader/writer for the data. Must be unmanaged and implement IElementReaderWriter{T} .

Type Constraints

TReaderWriter : *unmanaged*

TReaderWriter : [IElementReaderWriter](#)<*T*>

6.113.2 Member Function Documentation

6.113.2.1 Init()

```
override void Init (
    int capacity ) [virtual]
```

Initializes the [UnityValueSurrogate](#) with a specified capacity.

Parameters

<i>capacity</i>	The capacity to initialize the UnityValueSurrogate with.
-----------------	--

Implements [UnitySurrogateBase](#).

6.113.2.2 Read()

```
override void Read (
    int * data,
    int capacity ) [virtual]
```

Reads data into the [UnityValueSurrogate](#) from a specified memory location.

Parameters

<i>data</i>	The memory location to read from.
<i>capacity</i>	The number of elements to read.

Implements [UnitySurrogateBase](#).

6.113.2.3 Write()

```

override void Write (
    int * data,
    int capacity ) [virtual]

```

Writes data from the [UnityValueSurrogate](#) to a specified memory location.

Parameters

<i>data</i>	The memory location to write to.
<i>capacity</i>	The number of elements to write.

Implements [UnitySurrogateBase](#).

6.113.3 Property Documentation

6.113.3.1 DataProperty

```

abstract T DataProperty [get], [set]

```

Gets or sets the data for the [UnityValueSurrogate](#).

6.114 InterpolatedErrorCorrectionSettings Class Reference

A set of parameters that tune the interpolated correction of prediction error on transform data.

Public Attributes

- Single [MaxRate](#) = 10f
- Single [MinRate](#) = 3.3f
- Single [PosBlendEnd](#) = 1f
- Single [PosBlendStart](#) = 0.25f
- Single [PosMinCorrection](#) = 0.025f
- Single [PosTeleportDistance](#) = 2f
- Single [RotBlendEnd](#) = 0.5f
- Single [RotBlendStart](#) = 0.1f
- Single [RotTeleportRadians](#) = 1.5f

6.114.1 Detailed Description

A set of parameters that tune the interpolated correction of prediction error on transform data.

6.114.2 Member Data Documentation

6.114.2.1 MaxRate

Single MaxRate = 10f

A factor with dimension of 1/s (Hz) that works as a upper limit for how much of the accumulated prediction error is corrected every frame. This factor affects both the position and the rotation correction. Suggested values are greater than [MinRate](#) and smaller than half of a target rendering rate.

E.g.: MaxRate = 15, rendering delta time = (1/60)s: at maximum 25% ($15 * 1/60$) of the accumulated error will be corrected on this rendered frame.

This threshold might not be respected if the resultant correction magnitude is below the [PosMinCorrection](#) or above the [PosTeleportDistance](#), for the position error, or above the [RotTeleportRadians](#), for the rotation error.

6.114.2.2 MinRate

Single MinRate = 3.3f

A factor with dimension of 1/s (Hz) that works as a lower limit for how much of the accumulated prediction error is corrected every frame. This factor affects both the position and the rotation correction. Suggested values are greater than zero and smaller than [MaxRate](#).

E.g.: MinRate = 3, rendering delta time = (1/60)s: at least 5% ($3 * 1/60$) of the accumulated error will be corrected on this rendered frame.

This threshold might not be respected if the resultant correction magnitude is below the [PosMinCorrection](#) or above the [PosTeleportDistance](#), for the position error, or above the [RotTeleportRadians](#), for the rotation error.

6.114.2.3 PosBlendEnd

Single PosBlendEnd = 1f

The reference for the magnitude of the accumulated position error, in meters, at which the position error will be corrected at the [MaxRate](#). Suggested values are greater than [PosBlendStart](#) and smaller than [PosTeleportDistance](#).

In other words, if the magnitude of the accumulated error is equal to or greater than this threshold, it will be corrected at the [MaxRate](#). If, instead, the magnitude is between [PosBlendStart](#) and this threshold, the error is corrected at a rate between [MinRate](#) and [MaxRate](#), proportionally. If it is equal to or smaller than [PosBlendStart](#), it will be corrected at the [MinRate](#).

Note: as the factor is expressed in distance units (meters), it might need to be scaled proportionally to the overall scale of objects in the scene and speeds at which they move, which are factors that affect the expected magnitude of prediction errors.

6.114.2.4 PosBlendStart

```
Single PosBlendStart = 0.25f
```

The reference for the magnitude of the accumulated position error, in meters, at which the position error will be corrected at the [MinRate](#). Suggested values are greater than [PosMinCorrection](#) and smaller than [PosBlendEnd](#).

In other words, if the magnitude of the accumulated error is equal to or smaller than this threshold, it will be corrected at the [MinRate](#). If, instead, the magnitude is between this threshold and [PosBlendEnd](#), the error is corrected at a rate between [MinRate](#) and [MaxRate](#), proportionally. If it is equal to or greater than [PosBlendEnd](#), it will be corrected at the [MaxRate](#).

Note: as the factor is expressed in distance units (meters), it might need to be scaled proportionally to the overall scale of objects in the scene and speeds at which they move, which are factors that affect the expected magnitude of prediction errors.

6.114.2.5 PosMinCorrection

```
Single PosMinCorrection = 0.025f
```

The value, in meters, that represents the minimum magnitude of the accumulated position error that will be corrected in a single frame, until it is fully corrected.

This setting has priority over the resultant correction rate, i.e. the restriction will be respected even if it makes the effective correction rate be different than the one computed according to the min/max rates and start/end blend values. Suggested values are greater than zero and smaller than [PosBlendStart](#).

Note: as the factor is expressed in distance units (meters), it might need to be scaled proportionally to the overall scale of objects in the scene and speeds at which they move, which are factors that affect the expected magnitude of prediction errors.

6.114.2.6 PosTeleportDistance

```
Single PosTeleportDistance = 2f
```

The value, in meters, that represents the magnitude of the accumulated position error above which the error will be instantaneously corrected, effectively teleporting the rendered object to its correct position. Suggested values are greater than [PosBlendEnd](#).

This setting has priority over the resultant correction rate, i.e. the restriction will be respected even if it makes the effective correction rate be different than the one computed according to the min/max rates and start/end blend values.

Note: as the factor is expressed in distance units (meters), it might need to be scaled proportionally to the overall scale of objects in the scene and speeds at which they move, which are factors that affect the expected magnitude of prediction errors.

6.114.2.7 RotBlendEnd

```
Single RotBlendEnd = 0.5f
```

The reference for the magnitude of the accumulated rotation error, in radians, at which the rotation error will be corrected at the [MaxRate](#). Suggested values are greater than [RotBlendStart](#) and smaller than [RotTeleportRadians](#).

In other words, if the magnitude of the accumulated error is equal to or greater than this threshold, it will be corrected at the [MaxRate](#). If, instead, the magnitude is between [RotBlendStart](#) and this threshold, the error is corrected at a rate between [MinRate](#) and [MaxRate](#), proportionally. If it is equal to or smaller than [RotBlendStart](#), it will be corrected at the [MinRate](#).

6.114.2.8 RotBlendStart

```
Single RotBlendStart = 0.1f
```

The reference for the magnitude of the accumulated rotation error, in radians, at which the rotation error will be corrected at the [MinRate](#). Suggested values are smaller than [RotBlendEnd](#).

In other words, if the magnitude of the accumulated error is equal to or smaller than this threshold, it will be corrected at the [MinRate](#). If, instead, the magnitude is between this threshold and [RotBlendEnd](#), the error is corrected at a rate between [MinRate](#) and [MaxRate](#), proportionally. If it is equal to or greater than [RotBlendEnd](#), it will be corrected at the [MaxRate](#).

6.114.2.9 RotTeleportRadians

```
Single RotTeleportRadians = 1.5f
```

The value, in radians, that represents the magnitude of the accumulated rotation error above which the error will be instantaneously corrected, effectively teleporting the rendered object to its correct orientation. Suggested values are greater than [RotBlendEnd](#).

This setting has priority over the resultant correction rate, i.e. the restriction will be respected even if it makes the effective correction rate be different than the one computed according to the min/max rates and start/end blend values.

6.115 IPlayerJoined Interface Reference

Interface for handling the event when a player joins the game.

Inherits [IPublicFacingInterface](#).

Public Member Functions

- void [PlayerJoined](#) ([PlayerRef](#) player)
Method to be called when a player joins the game.

6.115.1 Detailed Description

Interface for handling the event when a player joins the game.

6.115.2 Member Function Documentation

6.115.2.1 PlayerJoined()

```
void PlayerJoined (  
    PlayerRef player )
```

Method to be called when a player joins the game.

Parameters

<i>player</i>	The player who joined the game.
---------------	---------------------------------

6.116 IPlayerLeft Interface Reference

Interface for handling the event when a player leaves the game.

Inherits [IPublicFacingInterface](#).

Public Member Functions

- void [PlayerLeft](#) ([PlayerRef](#) player)
Method to be called when a player leaves the game.

6.116.1 Detailed Description

Interface for handling the event when a player leaves the game.

6.116.2 Member Function Documentation

6.116.2.1 PlayerLeft()

```
void PlayerLeft (  
    PlayerRef player )
```

Method to be called when a player leaves the game.

Parameters

<i>player</i>	The player who left the game.
---------------	-------------------------------

6.117 IRemotePrefabCreated Interface Reference

Interface for handling the event when a remote prefab is created.

Inherits [IPublicFacingInterface](#).

Public Member Functions

- void [RemotePrefabCreated](#) ()
Method to be called when a remote prefab is created.

6.117.1 Detailed Description

Interface for handling the event when a remote prefab is created.

6.117.2 Member Function Documentation

6.117.2.1 RemotePrefabCreated()

```
void RemotePrefabCreated ( )
```

Method to be called when a remote prefab is created.

6.118 ISceneLoadDone Interface Reference

Interface for handling the event when a scene load operation is completed.

Inherits [IPublicFacingInterface](#).

Public Member Functions

- void [SceneLoadDone](#) (in [SceneLoadDoneArgs](#) sceneInfo)
Method to be called when a scene load operation is completed.

6.118.1 Detailed Description

Interface for handling the event when a scene load operation is completed.

6.118.2 Member Function Documentation

6.118.2.1 SceneLoadDone()

```
void SceneLoadDone (
    in SceneLoadDoneArgs sceneInfo )
```

Method to be called when a scene load operation is completed.

Parameters

<code>sceneInfo</code>	The information about the loaded scene.
------------------------	---

6.119 ISceneLoadStart Interface Reference

Interface for handling the event when a scene load operation is started.

Inherits [IPublicFacingInterface](#).

Public Member Functions

- void [SceneLoadStart](#) ([SceneRef](#) sceneRef)
Method to be called when a scene load operation is started.

6.119.1 Detailed Description

Interface for handling the event when a scene load operation is started.

6.119.2 Member Function Documentation

6.119.2.1 SceneLoadStart()

```
void SceneLoadStart (  
    SceneRef sceneRef )
```

Method to be called when a scene load operation is started.

Parameters

<code>sceneRef</code>	Reference to the scene that is being loaded.
-----------------------	--

6.120 ISimulationEnter Interface Reference

Interface for [SimulationEnter](#) callback. Called when the [NetworkObject](#) joins AreaOfInterest. Implement this interface on [SimulationBehaviour](#) and [NetworkBehaviour](#) classes.

Inherits [IPublicFacingInterface](#).

Public Member Functions

- void [SimulationEnter](#) ()

Called when the [NetworkObject](#) joins [AreaOfInterest](#). Object is now receiving snapshot updates. Object will execute [NetworkBehaviour](#) [FixedUpdateNetwork\(\)](#) and [Render\(\)](#) methods until the object leaves simulation.

6.120.1 Detailed Description

Interface for [SimulationEnter](#) callback. Called when the [NetworkObject](#) joins [AreaOfInterest](#). Implement this interface on [SimulationBehaviour](#) and [NetworkBehaviour](#) classes.

6.120.2 Member Function Documentation

6.120.2.1 SimulationEnter()

```
void SimulationEnter ( )
```

Called when the [NetworkObject](#) joins [AreaOfInterest](#). Object is now receiving snapshot updates. Object will execute [NetworkBehaviour](#) [FixedUpdateNetwork\(\)](#) and [Render\(\)](#) methods until the object leaves simulation.

6.121 ISimulationExit Interface Reference

Interface for the [SimulationExit](#) callback. Called when the [NetworkObject](#) leaves [AreaOfInterest](#). Implement this interface on [SimulationBehaviour](#) and [NetworkBehaviour](#) classes.

Inherits [IPublicFacingInterface](#).

Public Member Functions

- void [SimulationExit](#) ()

Called when the [NetworkObject](#) leaves [AreaOfInterest](#). Object is no longer receiving snapshot updates. Object will stop executing [NetworkBehaviour](#) [FixedUpdateNetwork\(\)](#) and [Render\(\)](#) methods until the object rejoins simulation.

6.121.1 Detailed Description

Interface for the [SimulationExit](#) callback. Called when the [NetworkObject](#) leaves [AreaOfInterest](#). Implement this interface on [SimulationBehaviour](#) and [NetworkBehaviour](#) classes.

6.121.2 Member Function Documentation

6.121.2.1 SimulationExit()

```
void SimulationExit ( )
```

Called when the [NetworkObject](#) leaves AreaOfInterest. Object is no longer receiving snapshot updates. Object will stop executing [NetworkBehaviour](#) FixedUpdateNetwork() and Render() methods until the object rejoins simulation.

6.122 ISpawned Interface Reference

Interface for handling the event when an object is spawned.

Inherits [IPublicFacingInterface](#).

Inherited by [HitboxManager](#), and [NetworkBehaviour](#).

Public Member Functions

- void [Spawned](#) ()
Method to be called when an object is spawned.

6.122.1 Detailed Description

Interface for handling the event when an object is spawned.

6.122.2 Member Function Documentation

6.122.2.1 Spawned()

```
void Spawned ( )
```

Method to be called when an object is spawned.

Implemented in [NetworkTransform](#), [NetworkMecanimAnimator](#), and [NetworkBehaviour](#).

6.123 IStateAuthorityChanged Interface Reference

Interface for handling the event when the state authority changes.

Inherits [IPublicFacingInterface](#).

Public Member Functions

- void [StateAuthorityChanged](#) ()
Method to be called when the state authority changes.

6.123.1 Detailed Description

Interface for handling the event when the state authority changes.

6.123.2 Member Function Documentation

6.123.2.1 StateAuthorityChanged()

```
void StateAuthorityChanged ( )
```

Method to be called when the state authority changes.

6.124 LagCompensatedHit Struct Reference

Defines a lag compensated query hit result.

Static Public Member Functions

- static [operator LagCompensatedHit](#) (RaycastHit raycastHit)
Creates a [LagCompensatedHit](#) structure from the information on a Unity RaycastHit.
- static [operator LagCompensatedHit](#) (RaycastHit2D raycastHit2D)
Creates a [LagCompensatedHit](#) structure from the information on a Unity RaycastHit2D.

Public Attributes

- Collider [Collider](#)
PhysX collider hit. Null in case hit is a [Fusion Hitbox](#) or a [Box2D](#) hit.
- Collider2D [Collider2D](#)
Box2D collider hit. Null in case hit is a [Fusion Hitbox](#) or a [PhysX](#) hit.
- float [Distance](#)
Distance (if requested) to hit, at the lag compensated time.
- GameObject [GameObject](#)
The Unity Game Object that was hit. Its data is not lag compensated. This is either the [Hitbox](#)'s or the [Collider](#)'s gameObject, depending on the object hit being a lag-compensated [Hitbox](#) or a regular Unity collider, respectively.
- [Hitbox](#) [Hitbox](#)
[Fusion](#)'s [Hitbox](#). Null in case the hit was on [PhysX](#) or [Box2D](#).
- Vector3 [HitboxColliderPosition](#)
The hitbox collider position on the snapshot.
- Quaternion [HitboxColliderRotation](#)
The hitbox collider rotation on the snapshot.
- Vector3 [Normal](#)
Surface normal (if requested) of the hit, at the lag compensated time.
- Vector3 [Point](#)
Point of impact of the hit, at the lag compensated time.
- [HitType](#) [Type](#)
Hit object source ([PhysX](#) or [Fusion](#) Hitboxes).

6.124.1 Detailed Description

Defines a lag compensated query hit result.

6.124.2 Member Function Documentation

6.124.2.1 operator LagCompensatedHit() [1/2]

```
static operator LagCompensatedHit (  
    RaycastHit raycastHit ) [explicit], [static]
```

Creates a [LagCompensatedHit](#) structure from the information on a Unity RaycastHit.

Parameters

<i>raycastHit</i>	The RaycastHit used as source.
-------------------	--------------------------------

Returns

The built [LagCompensatedHit](#) structure.

6.124.2.2 operator LagCompensatedHit() [2/2]

```
static operator LagCompensatedHit (  
    RaycastHit2D raycastHit2D ) [explicit], [static]
```

Creates a [LagCompensatedHit](#) structure from the information on a Unity RaycastHit2D.

Parameters

<i>raycastHit2D</i>	The RaycastHit2D used as source.
---------------------	----------------------------------

Returns

The built [LagCompensatedHit](#) structure.

6.124.3 Member Data Documentation

6.124.3.1 Collider

`Collider Collider`

PhysX collider hit. Null in case hit is a [Fusion Hitbox](#) or a Box2D hit.

6.124.3.2 Collider2D

`Collider2D Collider2D`

Box2D collider hit. Null in case hit is a [Fusion Hitbox](#) or a PhysX hit.

6.124.3.3 Distance

`float Distance`

Distance (if requested) to hit, at the lag compensated time.

6.124.3.4 GameObject

`GameObject GameObject`

The Unity Game Object that was hit. Its data is not lag compensated. This is either the [Hitbox's](#) or the [Collider's](#) `gameObject`, depending on the object hit being a lag-compensated [Hitbox](#) or a regular Unity collider, respectively.

6.124.3.5 Hitbox

[Hitbox Hitbox](#)

[Fusion's Hitbox](#). Null in case the hit was on PhysX or Box2D.

6.124.3.6 HitboxColliderPosition

`Vector3 HitboxColliderPosition`

The hitbox collider position on the snapshot.

6.124.3.7 HitboxColliderRotation

Quaternion HitboxColliderRotation

The hitbox collider rotation on the snapshot.

6.124.3.8 Normal

Vector3 Normal

Surface normal (if requested) of the hit, at the lag compensated time.

6.124.3.9 Point

Vector3 Point

Point of impact of the hit, at the lag compensated time.

6.124.3.10 Type

HitType Type

Hit object source (PhysX or [Fusion](#) Hitboxes).

6.125 AABB Struct Reference

Represents an Axis-Aligned Bounding Box ([AABB](#)).

Public Member Functions

- [AABB](#) (Bounds bounds)
Constructs an [AABB](#) from a Unity Bounds object.
- [AABB](#) (Vector3 center, Vector3 extents)
Constructs an [AABB](#) from a center point and extents.
- [AABB](#) (Vector3 center, Vector3 pointA, Vector3 pointB)
Constructs an [AABB](#) from a center point that encapsulate two other points.

Public Attributes

- readonly Vector3 [Center](#)
Represents the center of the [AABB](#).
- readonly Vector3 [Extents](#)
Represents the extents (half-widths) of the [AABB](#).
- readonly Vector3 [Max](#)
Represents the maximum point (upper corner) of the [AABB](#).
- readonly Vector3 [Min](#)
Represents the minimum point (lower corner) of the [AABB](#).

6.125.1 Detailed Description

Represents an Axis-Aligned Bounding Box ([AABB](#)).

6.125.2 Constructor & Destructor Documentation

6.125.2.1 [AABB\(\)](#) [1/3]

```
AABB (
    Bounds bounds )
```

Constructs an [AABB](#) from a Unity Bounds object.

Parameters

<i>bounds</i>	The Unity Bounds object to construct the AABB from.
---------------	---

6.125.2.2 [AABB\(\)](#) [2/3]

```
AABB (
    Vector3 center,
    Vector3 extents )
```

Constructs an [AABB](#) from a center point and extents.

Parameters

<i>center</i>	The center point of the AABB .
<i>extents</i>	The extents (half-widths) of the AABB .

6.125.2.3 AABB() [3/3]

```
AABB (
    Vector3 center,
    Vector3 pointA,
    Vector3 pointB )
```

Constructs an [AABB](#) from a center point that encapsulate two other points.

6.125.3 Member Data Documentation

6.125.3.1 Center

```
readonly Vector3 Center
```

Represents the center of the [AABB](#).

6.125.3.2 Extents

```
readonly Vector3 Extents
```

Represents the extents (half-widths) of the [AABB](#).

6.125.3.3 Max

```
readonly Vector3 Max
```

Represents the maximum point (upper corner) of the [AABB](#).

6.125.3.4 Min

```
readonly Vector3 Min
```

Represents the minimum point (lower corner) of the [AABB](#).

6.126 BoxOverlapQuery Class Reference

Class that represents a box overlap query. Used to query against the [NetworkRunner.LagCompensation](#) API.

Inherits [Query](#).

Public Member Functions

- [BoxOverlapQuery](#) (ref [BoxOverlapQueryParams](#) boxOverlapParams)
Create a new [BoxOverlapQuery](#) with the given *boxOverlapParams* .
- [BoxOverlapQuery](#) (ref [BoxOverlapQueryParams](#) boxOverlapParams, Collider[] physXOverlapHitsCache, Collider2D[] box2DOverlapHitsCache)
Create a new [BoxOverlapQuery](#) with the given *boxOverlapParams* . The result colliders arrays can be provided to avoid allocation.

Public Attributes

- Vector3 [Center](#)
The box query center.
- Vector3 [Extents](#)
The box query extents.
- Quaternion [Rotation](#)
The box query rotation.

Protected Member Functions

- override bool [Check](#) (ref [AABB](#) bounds)
Check if the given bounds overlaps with this query.

6.126.1 Detailed Description

Class that represents a box overlap query. Used to query against the [NetworkRunner.LagCompensation](#) API.

6.126.2 Constructor & Destructor Documentation

6.126.2.1 [BoxOverlapQuery\(\)](#) [1/2]

```
BoxOverlapQuery (
    ref BoxOverlapQueryParams boxOverlapParams )
```

Create a new [BoxOverlapQuery](#) with the given *boxOverlapParams* .

Parameters

<i>boxOverlapParams</i>	The parameters to be used when creating the query.
-------------------------	--

6.126.2.2 BoxOverlapQuery() [2/2]

```
BoxOverlapQuery (
    ref BoxOverlapQueryParams boxOverlapParams,
    Collider[] physXOverlapHitsCache,
    Collider2D[] box2DOverlapHitsCache )
```

Create a new [BoxOverlapQuery](#) with the given *boxOverlapParams* . The result colliders arrays can be provided to avoid allocation.

Parameters

<i>boxOverlapParams</i>	The parameters to be used when creating the query.
<i>physXOverlapHitsCache</i>	Array to write the results of the PhysX query if used.
<i>box2DOverlapHitsCache</i>	Array to write the results of the Box2D query if used.

6.126.3 Member Function Documentation

6.126.3.1 Check()

```
override bool Check (
    ref AABB bounds ) [protected], [virtual]
```

Check if the given *bounds* overlaps with this query.

Parameters

<i>bounds</i>	The bounds to check.
---------------	----------------------

Returns

True if the bounds overlaps with this query, false otherwise.

Implements [Query](#).

6.126.4 Member Data Documentation

6.126.4.1 Center

```
Vector3 Center
```

The box query center.

6.126.4.2 Extents

Vector3 Extents

The box query extents.

6.126.4.3 Rotation

Quaternion Rotation

The box query rotation.

6.127 BoxOverlapQueryParams Struct Reference

Base parameters needed to execute a box overlap query

Public Member Functions

- [BoxOverlapQueryParams](#) ([QueryParams](#) queryParams, Vector3 center, Vector3 extents, Quaternion rotation, int staticHitsCapacity)
Create a new [BoxOverlapQueryParams](#)

Public Attributes

- Vector3 [Center](#)
Represents the center of the box for the overlap query.
- Vector3 [Extents](#)
Represents the extents of the box for the overlap query.
- [QueryParams](#) [QueryParams](#)
Represents the base parameters for the query.
- Quaternion [Rotation](#)
Represents the rotation of the box for the overlap query.
- int [StaticHitsCapacity](#)
Represents the capacity for the cached PhysX and Box2D static hits.

6.127.1 Detailed Description

Base parameters needed to execute a box overlap query

6.127.2 Constructor & Destructor Documentation

6.127.2.1 BoxOverlapQueryParams()

```
BoxOverlapQueryParams (
    QueryParams queryParams,
    Vector3 center,
    Vector3 extents,
    Quaternion rotation,
    int staticHitsCapacity )
```

Create a new [BoxOverlapQueryParams](#)

Parameters

<i>queryParams</i>	Parameters to be used
<i>center</i>	The query center
<i>extents</i>	The query extents
<i>rotation</i>	The query rotation
<i>staticHitsCapacity</i>	Capacity for the cached PhysX and Box2D static hits.

6.127.3 Member Data Documentation

6.127.3.1 Center

Vector3 Center

Represents the center of the box for the overlap query.

6.127.3.2 Extents

Vector3 Extents

Represents the extents of the box for the overlap query.

6.127.3.3 QueryParams

[QueryParams](#) [QueryParams](#)

Represents the base parameters for the query.

6.127.3.4 Rotation

Quaternion Rotation

Represents the rotation of the box for the overlap query.

6.127.3.5 StaticHitsCapacity

```
int StaticHitsCapacity
```

Represents the capacity for the cached PhysX and Box2D static hits.

6.128 BVHDraw Class Reference

Provide a way to iterate over BVH and return a [BVHNodeDrawInfo](#) for each node.

Inherits [IEnumerable< BVHNodeDrawInfo >](#).

Public Member Functions

- [IEnumerable< BVHNodeDrawInfo > GetEnumerator \(\)](#)
Returns an enumerator that iterates through the [BVHNodeDrawInfo](#).

6.128.1 Detailed Description

Provide a way to iterate over BVH and return a [BVHNodeDrawInfo](#) for each node.

6.128.2 Member Function Documentation

6.128.2.1 GetEnumerator()

```
IEnumerable<BVHNodeDrawInfo> GetEnumerator \(\)
```

Returns an enumerator that iterates through the [BVHNodeDrawInfo](#).

6.129 BVHNodeDrawInfo Class Reference

Container class to provide the necessary info to draw nodes from the BVH

Properties

- [Bounds](#) [Bounds](#) [get]
Get the node Bounds
- [int Depth](#) [get]
Get the node depth on the BVH
- [int MaxDepth](#) [get]
Get the BVH max depth

6.129.1 Detailed Description

Container class to provide the necessary info to draw nodes from the BVH

6.129.2 Property Documentation

6.129.2.1 Bounds

Bounds Bounds [get]

Get the node Bounds

6.129.2.2 Depth

int Depth [get]

Get the node depth on the BVH

6.129.2.3 MaxDepth

int MaxDepth [get]

Get the BVH max depth

6.130 ColliderDrawInfo Class Reference

Container class to provide the necessary information to draw a hitbox collider

Properties

- Vector3 [BoxExtents](#) [get]
The box extends of the collider Used on [HitboxTypes](#) of types: Box
- Vector3 [CapsuleBottomCenter](#) [get]
Represents the bottom center position of the capsule collider.
- float [CapsuleExtents](#) [get]
*The height for capsule colliders.
See also*
[HitboxTypes](#)
- Vector3 [CapsuleTopCenter](#) [get]
Represents the top center position of the capsule collider.
- Matrix4x4 [LocalToWorldMatrix](#) [get]
The local to world matrix of the collider.
- Vector3 [Offset](#) [get]
The offset of the collider.
- float [Radius](#) [get]
The radius of the collider. Used on [HitboxTypes](#) of types: Sphere and Capsule.
- [HitboxTypes Type](#) [get]
The [HitboxTypes](#) of the collider.

6.130.1 Detailed Description

Container class to provide the necessary information to draw a hitbox collider

6.130.2 Property Documentation

6.130.2.1 BoxExtents

`Vector3 BoxExtents [get]`

The box extends of the collider Used on [HitboxTypes](#) of types: Box

6.130.2.2 CapsuleBottomCenter

`Vector3 CapsuleBottomCenter [get]`

Represents the bottom center position of the capsule collider.

6.130.2.3 CapsuleExtents

`float CapsuleExtents [get]`

The height for capsule colliders.

See also

[HitboxTypes](#)

6.130.2.4 CapsuleTopCenter

`Vector3 CapsuleTopCenter [get]`

Represents the top center position of the capsule collider.

6.130.2.5 LocalToWorldMatrix

`Matrix4x4 LocalToWorldMatrix [get]`

The local to world matrix of the collider.

6.130.2.6 Offset

`Vector3 Offset [get]`

The offset of the collider.

6.130.2.7 Radius

`float Radius [get]`

The radius of the collider. Used on [HitboxTypes](#) of types: Sphere and Capsule.

6.130.2.8 Type

`HitboxTypes Type [get]`

The [HitboxTypes](#) of the collider.

6.131 HitboxColliderContainerDraw Class Reference

Provide a way to iterate over the `HitboxBuffer.HitboxSnapshot` and return the [ColliderDrawInfo](#) for each collider on the snapshot.

Inherits `IEnumerable< ColliderDrawInfo >`.

Public Member Functions

- `IEnumerator< ColliderDrawInfo > GetEnumerator ()`
Returns an enumerator that iterates through the [ColliderDrawInfo](#) of this container.

6.131.1 Detailed Description

Provide a way to iterate over the `HitboxBuffer.HitboxSnapshot` and return the [ColliderDrawInfo](#) for each collider on the snapshot.

6.131.2 Member Function Documentation

6.131.2.1 GetEnumerator()

```
IEnumerator<ColliderDrawInfo> GetEnumerator ( )
```

Returns an enumerator that iterates through the [ColliderDrawInfo](#) of this container.

6.132 LagCompensatedExt Class Reference

LagCompensated Extension methods

Static Public Member Functions

- static void [SortDistance](#) (this List< [LagCompensatedHit](#) > hits)
Sorts all hits in ascending order of [LagCompensatedHit.Distance](#).
- static void [SortReference](#) (this List< [LagCompensatedHit](#) > hits, Vector3 reference)
Sorts all hits in ascending order of distance from [LagCompensatedHit.Point](#) to the reference point.

6.132.1 Detailed Description

LagCompensated Extension methods

6.132.2 Member Function Documentation

6.132.2.1 SortDistance()

```
static void SortDistance (
    this List< LagCompensatedHit > hits ) [static]
```

Sorts all *hits* in ascending order of [LagCompensatedHit.Distance](#).

Parameters

<i>hits</i>	List containing hits to be sorted.
-------------	------------------------------------

Exceptions

<i>NullReferenceException</i>	If <i>hits</i> are null.
-------------------------------	--------------------------

6.132.2.2 SortReference()

```
static void SortReference (
    this List< LagCompensatedHit > hits,
    Vector3 reference ) [static]
```

Sorts all *hits* in ascending order of distance from [LagCompensatedHit.Point](#) to the *reference* point.

Parameters

<i>hits</i>	List containing hits to be sorted.
<i>reference</i>	Used as reference point to compute distance from hit points.

Exceptions

<i>NullReferenceException</i>	If <i>hits</i> are null.
-------------------------------	--------------------------

6.133 LagCompensationDraw Class Reference

Provide access to iterate over the lag compensation system components and give the necessary information to draw them.

Static Public Member Functions

- static void [GizmosDrawWireCapsule](#) (Vector3 topCenter, Vector3 bottomCenter, float capsuleRadius)
Method to draw capsules out of simple shapes.

Public Attributes

- [BVHDraw BVHDraw](#)
Iterate over to get the BVH node draw data.
- [SnapshotHistoryDraw SnapshotHistoryDraw](#)
Iterate over to get the hitbox snapshots draw data. Iterate the received hitbox snapshot draw data to get all the colliders draw info for that snapshot.

6.133.1 Detailed Description

Provide access to iterate over the lag compensation system components and give the necessary information to draw them.

6.133.2 Member Function Documentation

6.133.2.1 GizmosDrawWireCapsule()

```
static void GizmosDrawWireCapsule (
    Vector3 topCenter,
    Vector3 bottomCenter,
    float capsuleRadius ) [static]
```

Method to draw capsules out of simple shapes.

Parameters

<i>topCenter</i>	The top capsule end position
<i>bottomCenter</i>	The bottom capsule end position
<i>capsuleRadius</i>	The capsule radius

6.133.3 Member Data Documentation

6.133.3.1 BVHDraw

[BVHDraw](#) [BVHDraw](#)

Iterate over to get the BVH node draw data.

6.133.3.2 SnapshotHistoryDraw

[SnapshotHistoryDraw](#) [SnapshotHistoryDraw](#)

Iterate over to get the hitbox snapshots draw data. Iterate the received hitbox snapshot draw data to get all the colliders draw info for that snapshot.

6.134 LagCompensationUtils.ContactData Struct Reference

Details regarding a shape intersection. It does not carry information about the intersection happening or not.

Public Attributes

- Vector3 [Normal](#)
Vector that described the plane of smallest penetration between the shapes.
- float [Penetration](#)
Penetration along the normal plane.
- Vector3 [Point](#)
Contact point.

6.134.1 Detailed Description

Details regarding a shape intersection. It does not carry information about the intersection happening or not.

6.134.2 Member Data Documentation

6.134.2.1 Normal

Vector3 Normal

Vector that described the plane of smallest penetration between the shapes.

6.134.2.2 Penetration

float Penetration

Penetration along the normal plane.

6.134.2.3 Point

Vector3 Point

Contact point.

6.135 PositionRotationQueryParams Struct Reference

[Query](#) parameters for position rotation query

Public Member Functions

- [PositionRotationQueryParams](#) ([QueryParams](#) queryParams, [Hitbox](#) hitbox)
Create a new [PositionRotationQueryParams](#).

Public Attributes

- [Hitbox](#) Hitbox
Represents the hitbox to be queried.
- [QueryParams](#) QueryParams
Represents the base parameters for the query.

6.135.1 Detailed Description

[Query](#) parameters for position rotation query

6.135.2 Constructor & Destructor Documentation

6.135.2.1 PositionRotationQueryParams()

```
PositionRotationQueryParams (
    QueryParams queryParams,
    Hitbox hitbox )
```

Create a new [PositionRotationQueryParams](#).

Parameters

<i>queryParams</i>	Parameters to be used
<i>hitbox</i>	The hitbox to be queried

6.135.3 Member Data Documentation

6.135.3.1 Hitbox

[Hitbox](#) Hitbox

Represents the hitbox to be queried.

6.135.3.2 QueryParams

[QueryParams](#) QueryParams

Represents the base parameters for the query.

6.136 Query Class Reference

Base class for all Lag Compensation queries

Inherits [IBoundsTraversalTest](#).

Inherited by [BoxOverlapQuery](#), [RaycastQuery](#), and [SphereOverlapQuery](#).

Public Attributes

- float? [Alpha](#)
Represents the interpolation factor between the current and next simulation tick.
- LayerMask [LayerMask](#)
Represents the layer mask to selectively ignore colliders when performing the query.
- [HitOptions](#) [Options](#)
Represents the options for the hit detection of the query.
- [PlayerRef](#) [Player](#)
Represents the player who initiated the query.
- [PreProcessingDelegate](#) [PreProcessingDelegate](#)
Represents the delegate to be called for pre-processing before the query is performed.
- int? [Tick](#)
Represents the simulation tick at which the query was initiated.
- int? [TickTo](#)
Represents the simulation tick to which the query is performed.
- [QueryTriggerInteraction](#) [TriggerInteraction](#)
Represents the interaction type of the query with triggers.
- void * [UserArgs](#)
Represents the user arguments for the query.

Protected Member Functions

- abstract bool [Check](#) (ref [AABB](#) bounds)
Checks if the provided bounds should be included in the query.
- [Query](#) (ref [QueryParams](#) qParams)
Initializes a new instance of the [Query](#) class using the provided [QueryParams](#).

6.136.1 Detailed Description

Base class for all Lag Compensation queries

6.136.2 Constructor & Destructor Documentation

6.136.2.1 [Query\(\)](#)

```
Query (
    ref QueryParams qParams ) [protected]
```

Initializes a new instance of the [Query](#) class using the provided [QueryParams](#).

Parameters

<i>qParams</i>	The QueryParams to use for initializing the Query .
----------------	---

6.136.3 Member Function Documentation

6.136.3.1 Check()

```
abstract bool Check (  
    ref AABB bounds ) [protected], [pure virtual]
```

Checks if the provided bounds should be included in the query.

Parameters

<i>bounds</i>	The bounds to check.
---------------	----------------------

Returns

True if the bounds should be included in the query, false otherwise.

Implemented in [SphereOverlapQuery](#), [RaycastQuery](#), and [BoxOverlapQuery](#).

6.136.4 Member Data Documentation

6.136.4.1 Alpha

```
float? Alpha
```

Represents the interpolation factor between the current and next simulation tick.

6.136.4.2 LayerMask

```
LayerMask LayerMask
```

Represents the layer mask to selectively ignore colliders when performing the query.

6.136.4.3 Options

```
HitOptions Options
```

Represents the options for the hit detection of the query.

6.136.4.4 Player

`PlayerRef` `Player`

Represents the player who initiated the query.

6.136.4.5 PreProcessingDelegate

`PreProcessingDelegate` `PreProcessingDelegate`

Represents the delegate to be called for pre-processing before the query is performed.

6.136.4.6 Tick

`int?` `Tick`

Represents the simulation tick at which the query was initiated.

6.136.4.7 TickTo

`int?` `TickTo`

Represents the simulation tick to which the query is performed.

6.136.4.8 TriggerInteraction

`QueryTriggerInteraction` `TriggerInteraction`

Represents the interaction type of the query with triggers.

6.136.4.9 UserArgs

`void*` `UserArgs`

Represents the user arguments for the query.

6.137 QueryParams Struct Reference

Base parameters needed to execute a query.

Public Attributes

- float? [Alpha](#)
Represents the interpolation factor between the current and next simulation tick.
- LayerMask [LayerMask](#)
Represents the layer mask to selectively ignore colliders when performing the query.
- [HitOptions](#) [Options](#)
Represents the options for the hit detection of the query.
- [PlayerRef](#) [Player](#)
Represents the player who initiated the query.
- [PreProcessingDelegate](#) [PreProcessingDelegate](#)
Represents the delegate to be called for pre-processing before the query is performed.
- int [Tick](#)
Represents the simulation tick at which the query was initiated.
- int? [TickTo](#)
Represents the simulation tick to which the query is performed.
- [QueryTriggerInteraction](#) [TriggerInteraction](#)
Represents the interaction type of the query with triggers.
- void * [UserArgs](#)
Represents the user arguments for the query.

6.137.1 Detailed Description

Base parameters needed to execute a query.

6.137.2 Member Data Documentation

6.137.2.1 Alpha

float? Alpha

Represents the interpolation factor between the current and next simulation tick.

6.137.2.2 LayerMask

LayerMask LayerMask

Represents the layer mask to selectively ignore colliders when performing the query.

6.137.2.3 Options

`HitOptions` Options

Represents the options for the hit detection of the query.

6.137.2.4 Player

`PlayerRef` Player

Represents the player who initiated the query.

6.137.2.5 PreProcessingDelegate

`PreProcessingDelegate` PreProcessingDelegate

Represents the delegate to be called for pre-processing before the query is performed.

6.137.2.6 Tick

`int` Tick

Represents the simulation tick at which the query was initiated.

6.137.2.7 TickTo

`int?` TickTo

Represents the simulation tick to which the query is performed.

6.137.2.8 TriggerInteraction

`QueryTriggerInteraction` TriggerInteraction

Represents the interaction type of the query with triggers.

6.137.2.9 UserArgs

```
void* UserArgs
```

Represents the user arguments for the query.

6.138 RaycastAllQuery Class Reference

Class that represents a raycast all query. Used to query against the [NetworkRunner.LagCompensation](#) API.

Inherits [RaycastQuery](#).

Public Member Functions

- [RaycastAllQuery](#) (ref [RaycastQueryParams](#) raycastQueryParams)
Create a new [RaycastAllQuery](#) with the given [RaycastQueryParams](#).
- [RaycastAllQuery](#) (ref [RaycastQueryParams](#) raycastQueryParams, RaycastHit[] physXRaycastHitsCache, RaycastHit2D[] box2DRaycastHitCache)
Create a new [RaycastAllQuery](#) with the given [RaycastQueryParams](#). The result colliders arrays can be provided to avoid allocation.

Additional Inherited Members

6.138.1 Detailed Description

Class that represents a raycast all query. Used to query against the [NetworkRunner.LagCompensation](#) API.

6.138.2 Constructor & Destructor Documentation

6.138.2.1 RaycastAllQuery() [1/2]

```
RaycastAllQuery (
    ref RaycastQueryParams raycastQueryParams )
```

Create a new [RaycastAllQuery](#) with the given [RaycastQueryParams](#).

Parameters

<code>raycastQueryParams</code>	The parameters to be used when creating the query.
---------------------------------	--

6.138.2.2 RaycastAllQuery() [2/2]

```
RaycastAllQuery (
    ref RaycastQueryParams raycastQueryParams,
    RaycastHit[] physXRaycastHitsCache,
    RaycastHit2D[] box2DRaycastHitCache )
```

Create a new [RaycastAllQuery](#) with the given [RaycastQueryParams](#). The result colliders arrays can be provided to avoid allocation.

Parameters

<code>raycastQueryParams</code>	The parameters to be used when creating the query.
<code>physXRaycastHitsCache</code>	Array to write the results of the PhysX query if used.
<code>box2DRaycastHitCache</code>	Array to write the results of the Box2D query if used.

6.139 RaycastQuery Class Reference

Class that represents a raycast query. Used to query against the [NetworkRunner.LagCompensation](#) API.

Inherits [Query](#).

Inherited by [RaycastAllQuery](#).

Public Member Functions

- [RaycastQuery](#) (ref [RaycastQueryParams](#) raycastQueryParams)
Create a new [RaycastQuery](#) with the given [RaycastQueryParams](#)

Public Attributes

- Vector3 [Direction](#)
Represents the direction of the raycast for the query.
- float [Length](#)
Represents the maximum length of the raycast for the query.
- Vector3 [Origin](#)
Represents the origin point of the raycast for the query.

Protected Member Functions

- override bool [Check](#) (ref [AABB](#) bounds)
Check if the provided bounds should be included in the query.

6.139.1 Detailed Description

Class that represents a raycast query. Used to query against the [NetworkRunner.LagCompensation](#) API.

6.139.2 Constructor & Destructor Documentation

6.139.2.1 RaycastQuery()

```
RaycastQuery (
    ref RaycastQueryParams raycastQueryParams )
```

Create a new [RaycastQuery](#) with the given [RaycastQueryParams](#)

Parameters

<i>raycastQueryParams</i>	The parameters to be used when creating the query.
---------------------------	--

6.139.3 Member Function Documentation

6.139.3.1 Check()

```
override bool Check (
    ref AABB bounds ) [protected], [virtual]
```

Check if the provided bounds should be included in the query.

Parameters

<i>bounds</i>	The bounds to check.
---------------	----------------------

Returns

True if the bounds should be included in the query, false otherwise.

Implements [Query](#).

6.139.4 Member Data Documentation

6.139.4.1 Direction

```
Vector3 Direction
```

Represents the direction of the raycast for the query.

6.139.4.2 Length

float Length

Represents the maximum length of the raycast for the query.

6.139.4.3 Origin

Vector3 Origin

Represents the origin point of the raycast for the query.

6.140 RaycastQueryParams Struct Reference

Base parameters needed to execute a raycast query

Public Member Functions

- [RaycastQueryParams](#) ([QueryParams](#) queryParams, [Vector3](#) origin, [Vector3](#) direction, float length, int staticHitsCapacity=64)
Create a new [RaycastQueryParams](#)

Public Attributes

- [Vector3](#) [Direction](#)
Represents the direction of the raycast for the query.
- float [Length](#)
Represents the maximum length of the raycast for the query.
- [Vector3](#) [Origin](#)
Represents the origin point of the raycast for the query.
- [QueryParams](#) [QueryParams](#)
Represents the base parameters for the raycast query.
- int [StaticHitsCapacity](#)
Represents the capacity for the cached PhysX and Box2D static hits.

6.140.1 Detailed Description

Base parameters needed to execute a raycast query

6.140.2 Constructor & Destructor Documentation

6.140.2.1 RaycastQueryParams()

```
RaycastQueryParams (
    QueryParams queryParams,
    Vector3 origin,
    Vector3 direction,
    float length,
    int staticHitsCapacity = 64 )
```

Create a new [RaycastQueryParams](#)

Parameters

<i>queryParams</i>	Parameters to be used
<i>origin</i>	The raycast origin
<i>direction</i>	The raycast direction
<i>length</i>	The raycast max length
<i>staticHitsCapacity</i>	Capacity for the cached PhysX and Box2D static hits.

6.140.3 Member Data Documentation**6.140.3.1 Direction**

Vector3 Direction

Represents the direction of the raycast for the query.

6.140.3.2 Length

float Length

Represents the maximum length of the raycast for the query.

6.140.3.3 Origin

Vector3 Origin

Represents the origin point of the raycast for the query.

6.140.3.4 QueryParams

[QueryParams](#) [QueryParams](#)

Represents the base parameters for the raycast query.

6.140.3.5 StaticHitsCapacity

```
int StaticHitsCapacity
```

Represents the capacity for the cached PhysX and Box2D static hits.

6.141 SnapshotHistoryDraw Class Reference

Provide a way to iterate over the HitboxBuffer and return the [HitboxColliderContainerDraw](#) container for each snapshot on the buffer.

Inherits [IEnumerable< HitboxColliderContainerDraw >](#).

Public Member Functions

- [IEnumerator< HitboxColliderContainerDraw > GetEnumerator \(\)](#)
Returns an enumerator that iterates through the [HitboxColliderContainerDraw](#).

6.141.1 Detailed Description

Provide a way to iterate over the HitboxBuffer and return the [HitboxColliderContainerDraw](#) container for each snapshot on the buffer.

6.141.2 Member Function Documentation

6.141.2.1 GetEnumerator()

```
IEnumerator<HitboxColliderContainerDraw> GetEnumerator ()
```

Returns an enumerator that iterates through the [HitboxColliderContainerDraw](#).

6.142 SphereOverlapQuery Class Reference

Class that represents a sphere overlap query. Used to query against the [NetworkRunner.LagCompensation](#) API.

Inherits [Query](#).

Public Member Functions

- [SphereOverlapQuery](#) (ref [SphereOverlapQueryParams](#) sphereOverlapParams)
Create a new [SphereOverlapQuery](#) with the given [SphereOverlapQueryParams](#).
- [SphereOverlapQuery](#) (ref [SphereOverlapQueryParams](#) sphereOverlapParams, Collider[] physXOverlapHitsCache, Collider2D[] box2DOverlapHitsCache)
Create a new [SphereOverlapQuery](#) with the given [SphereOverlapQueryParams](#).

Public Attributes

- Vector3 [Center](#)
Represents the center of the sphere for the overlap query.
- float [Radius](#)
Represents the radius of the sphere for the overlap query.

Protected Member Functions

- override bool [Check](#) (ref [AABB](#) bounds)
Check if the given [AABB](#) intersects with the sphere overlap query.

6.142.1 Detailed Description

Class that represents a sphere overlap query. Used to query against the [NetworkRunner.LagCompensation](#) API.

6.142.2 Constructor & Destructor Documentation

6.142.2.1 SphereOverlapQuery() [1/2]

```
SphereOverlapQuery (
    ref SphereOverlapQueryParams sphereOverlapParams )
```

Create a new [SphereOverlapQuery](#) with the given [SphereOverlapQueryParams](#).

Parameters

<code>sphereOverlapParams</code>	The parameters to be used when creating the query.
----------------------------------	--

6.142.2.2 SphereOverlapQuery() [2/2]

```
SphereOverlapQuery (
    ref SphereOverlapQueryParams sphereOverlapParams,
    Collider[] physXOverlapHitsCache,
    Collider2D[] box2DOverlapHitsCache )
```

Create a new [SphereOverlapQuery](#) with the given [SphereOverlapQueryParams](#).

Parameters

<code>sphereOverlapParams</code>	The parameters to be used when creating the query.
<code>physXOverlapHitsCache</code>	Array to write the results of the PhysX query if used.
<code>box2DOverlapHitsCache</code>	Array to write the results of the Box2D query if used.

6.142.3 Member Function Documentation

6.142.3.1 Check()

```
override bool Check (  
    ref AABB bounds ) [protected], [virtual]
```

Check if the given [AABB](#) intersects with the sphere overlap query.

Parameters

<i>bounds</i>	The bounds to check against.
---------------	------------------------------

Returns

True if the bounds intersects with the sphere overlap query, false otherwise.

Implements [Query](#).

6.142.4 Member Data Documentation

6.142.4.1 Center

```
Vector3 Center
```

Represents the center of the sphere for the overlap query.

6.142.4.2 Radius

```
float Radius
```

Represents the radius of the sphere for the overlap query.

6.143 SphereOverlapQueryParams Struct Reference

Base parameters needed to execute a sphere overlap query

Public Member Functions

- [SphereOverlapQueryParams](#) ([QueryParams](#) queryParams, [Vector3](#) center, float radius, int staticHits↔ Capacity)

Create a new [SphereOverlapQueryParams](#).

Public Attributes

- [Vector3](#) [Center](#)
Represents the center of the sphere for the overlap query.
- [QueryParams](#) [QueryParams](#)
Represents the base parameters for the sphere overlap query.
- float [Radius](#)
Represents the radius of the sphere for the overlap query.
- int [StaticHitsCapacity](#)
Represents the capacity for the cached PhysX and Box2D static hits.

6.143.1 Detailed Description

Base parameters needed to execute a sphere overlap query

6.143.2 Constructor & Destructor Documentation

6.143.2.1 SphereOverlapQueryParams()

```
SphereOverlapQueryParams (
    QueryParams queryParams,
    Vector3 center,
    float radius,
    int staticHitsCapacity )
```

Create a new [SphereOverlapQueryParams](#).

Parameters

<i>queryParams</i>	Parameters to be used
<i>center</i>	The query center
<i>radius</i>	The query radius
<i>staticHitsCapacity</i>	Capacity for the cached PhysX and Box2D static hits.

6.143.3 Member Data Documentation

6.143.3.1 Center

Vector3 Center

Represents the center of the sphere for the overlap query.

6.143.3.2 QueryParams

QueryParams QueryParams

Represents the base parameters for the sphere overlap query.

6.143.3.3 Radius

float Radius

Represents the radius of the sphere for the overlap query.

6.143.3.4 StaticHitsCapacity

int StaticHitsCapacity

Represents the capacity for the cached PhysX and Box2D static hits.

6.144 LagCompensationSettings Class Reference

Settings for lag compensation history.

Public Attributes

- int [CachedStaticCollidersSize](#) = 64
The size of the cached static colliders (PhysX or Box2D) array of the default Lag Compensation Queries.
- bool [Enabled](#) = false
Indicates if a [HitboxManager](#) instance should be added when the [NetworkRunner](#) is initialized.
- int [HitboxBufferLengthInMs](#) = 200
Hitbox snapshot history length in milliseconds.
- int [HitboxDefaultCapacity](#) = 512
Hitbox capacity per snapshot.

Properties

- float [ExpansionFactor](#) [get]
Broadphase BVH node expansion factor (default 20%) for leaf nodes, so updates are not too frequent.
- bool [Optimize](#) [get]
Optional: tries to optimize broadphase BVH every update. May be removed in the future.

6.144.1 Detailed Description

Settings for lag compensation history.

6.144.2 Member Data Documentation

6.144.2.1 CachedStaticCollidersSize

```
int CachedStaticCollidersSize = 64
```

The size of the cached static colliders (PhysX or Box2D) array of the default Lag Compensation Queries.

6.144.2.2 Enabled

```
bool Enabled = false
```

Indicates if a [HitboxManager](#) instance should be added when the [NetworkRunner](#) is initialized.

6.144.2.3 HitboxBufferLengthInMs

```
int HitboxBufferLengthInMs = 200
```

[Hitbox](#) snapshot history length in milliseconds.

6.144.2.4 HitboxDefaultCapacity

```
int HitboxDefaultCapacity = 512
```

[Hitbox](#) capacity per snapshot.

6.144.3 Property Documentation

6.144.3.1 ExpansionFactor

```
float ExpansionFactor [get]
```

Broadphase BVH node expansion factor (default 20%) for leaf nodes, so updates are not too frequent.

6.144.3.2 Optimize

```
bool Optimize [get]
```

Optional: tries to optimize broadphase BVH every update. May be removed in the future.

6.145 LayerAttribute Class Reference

Specifies that an int field should be drawn as a layer field in the inspector.

Inherits [DrawerPropertyAttribute](#).

6.145.1 Detailed Description

Specifies that an int field should be drawn as a layer field in the inspector.

6.146 LayerMatrixAttribute Class Reference

Specifies that the integer array field should be drawn as a layer matrix in the inspector.

Inherits [DrawerPropertyAttribute](#).

6.146.1 Detailed Description

Specifies that the integer array field should be drawn as a layer matrix in the inspector.

6.147 LobbyInfo Class Reference

Holds information about a Lobby

Properties

- bool [IsValid](#) [get]
Flag to signal if the [LobbyInfo](#) is ready for use. This is only true if the peer is currently connected to a Lobby.
- string [Name](#) [get]
Lobby Name
- string [Region](#) [get]
Stores the current connected Region

6.147.1 Detailed Description

Holds information about a Lobby

6.147.2 Property Documentation

6.147.2.1 IsValid

```
bool IsValid [get]
```

Flag to signal if the [LobbyInfo](#) is ready for use. This is only true if the peer is currently connected to a Lobby.

6.147.2.2 Name

```
string Name [get]
```

Lobby Name

6.147.2.3 Region

```
string Region [get]
```

Stores the current connected Region

6.148 Mask256 Struct Reference

[Mask256](#) is a 256-bit mask that can be used to store 256 boolean values.

Inherits `IEquatable< Mask256 >`.

Public Member Functions

- void [Clear](#) ()
Sets all bits to 0.
- bool [Equals](#) ([Mask256](#) other)
Returns true if the masks are equal.
- override bool [Equals](#) (object obj)
Returns true if the masks are equal.
- bool [GetBit](#) (int bitIndex)
Returns a specific bit from the mask.
- override int [GetHashCode](#) ()
Calculates the hash code of the mask.
- bool [IsNothing](#) ()
Returns true if the mask is empty.
- [Mask256](#) (long a, long b=0, long c=0, long d=0)
Creates a new mask with specified initial values.
- void [SetBit](#) (int bitIndex, bool set)
Sets a specific bit in the mask.
- override string [ToString](#) ()
Converts the mask to a string in the form of "a:b:c:d".

Static Public Member Functions

- static implicit [operator long](#) ([Mask256](#) mask)
Equivalent to
- static implicit [operator Mask256](#) (long value)
Converts a long value to a mask.
- static [Mask256 operator&](#) ([Mask256](#) a, [Mask256](#) b)
Performs a logical-AND operation.
- static [Mask256 operator|](#) ([Mask256](#) a, [Mask256](#) b)
Performs a logical-OR operation.
- static [Mask256 operator~](#) ([Mask256](#) a)
Performs a logical-NOT operation.

Properties

- long [this\[int i\]](#) [get, set]
Returns selected 64-bit value from the mask.

6.148.1 Detailed Description

[Mask256](#) is a 256-bit mask that can be used to store 256 boolean values.

6.148.2 Constructor & Destructor Documentation

6.148.2.1 Mask256()

```
Mask256 (
    long a,
    long b = 0,
    long c = 0,
    long d = 0 )
```

Creates a new mask with specified initial values.

6.148.3 Member Function Documentation

6.148.3.1 Clear()

```
void Clear ( )
```

Sets all bits to 0.

6.148.3.2 Equals() [1/2]

```
bool Equals (
    Mask256 other )
```

Returns `true` if the masks are equal.

6.148.3.3 Equals() [2/2]

```
override bool Equals (
    object obj )
```

Returns `true` if the masks are equal.

6.148.3.4 GetBit()

```
bool GetBit (
    int bitIndex )
```

Returns a specific bit from the mask.

6.148.3.5 GetHashCode()

```
override int GetHashCode ( )
```

Calculates the hash code of the mask.

6.148.3.6 IsNothing()

```
bool IsNothing ( )
```

Returns `true` if the mask is empty.

Returns

6.148.3.7 operator long()

```
static implicit operator long (
    Mask256 mask ) [static]
```

Equivalent to

```
mask[0]
```

6.148.3.8 operator Mask256()

```
static implicit operator Mask256 (
    long value ) [static]
```

Converts a long value to a mask.

6.148.3.9 operator&()

```
static Mask256 operator& (
    Mask256 a,
    Mask256 b ) [static]
```

Performs a logical-AND operation.

6.148.3.10 operator|()

```
static Mask256 operator| (
    Mask256 a,
    Mask256 b ) [static]
```

Performs a logical-OR operation.

6.148.3.11 operator~()

```
static Mask256 operator~ (
    Mask256 a ) [static]
```

Performs a logical-NOT operation.

6.148.3.12 SetBit()

```
void SetBit (
    int bitIndex,
    bool set )
```

Sets a specific bit in the mask.

6.148.3.13 ToString()

```
override string ToString ( )
```

Converts the mask to a string in the form of "a:b:c:d".

6.148.4 Property Documentation

6.148.4.1 this[int i]

```
long this[int i] [get], [set]
```

Returns selected 64-bit value from the mask.

Parameters

<i>i</i>	
----------	--

6.149 Maths Class Reference

Math utility methods.

Classes

- struct [FastAbs](#)

Represents a structure that allows quick setting of the sign bit of floats using field offsets.

Static Public Member Functions

- static int [BitScanReverse](#) (int v)
Finds the index of the most significant set bit (1-bit) in a 32-bit integer.
- static int [BitScanReverse](#) (long v)
Finds the index of the most significant set bit (1-bit) in a 64-bit integer.
- static int [BitScanReverse](#) (uint v)
Finds the index of the most significant set bit (1-bit) in a 32-bit unsigned integer.
- static int [BitScanReverse](#) (ulong v)
Finds the index of the most significant set bit (1-bit) in a 64-bit unsigned integer.
- static int [BitsRequiredForNumber](#) (int n)
Determines the number of bits required to represent the specified integer.
- static int [BitsRequiredForNumber](#) (uint n)
Determines the number of bits required to represent a uint value.
- static int [BytesRequiredForBits](#) (int b)
Calculates the number of bytes required to store the specified number of bits.
- static short [BytesRequiredForBits](#) (short b)
Calculates the number of bytes required to store the specified number of bits.
- static int [CeilToInt](#) (double value)
Rounds a double value up to the nearest integer.
- static double [Clamp](#) (double v, double min, double max)
Clamps a double value to the specified range.
- static float [Clamp](#) (float v, float min, float max)
Clamps a float value to the specified range.
- static int [Clamp](#) (int v, int min, int max)
Clamps an integer value to the specified range.
- static uint [Clamp](#) (uint v, uint min, uint max)
Clamps an unsigned integer value to the specified range.
- static double [Clamp01](#) (double v)
Clamps a double value to the range [0, 1].
- static float [Clamp01](#) (float v)
Clamps a float value to the range [0, 1].
- static byte [ClampToByte](#) (int v)
Clamps an integer value to the range [0, 255] and returns it as a byte.
- static double [CosineInterpolate](#) (double a, double b, double t)
Performs cosine interpolation between two values.
- static int [CountSetBits](#) (ulong x)
Counts the number of bits set to 1 in a ulong value.
- static int [CountUsedBitsMinOne](#) (uint value)
Counts the number of bits set to 1 in a uint value, minus one.

- static int [FloorToInt](#) (double value)
Rounds a double value down to the nearest integer.
- static int [IntsRequiredForBits](#) (int b)
Calculates the number of integers required to store the specified number of bits.
- static double [Lerp](#) (double a, double b, double t)
Linearly interpolates between two double values.
- static float [Lerp](#) (float a, float b, float t)
Linearly interpolates between two float values.
- static double [MicrosecondsToSeconds](#) (long microseconds)
Converts microseconds to seconds.
- static long [MillisecondsToMicroseconds](#) (long milliseconds)
Converts milliseconds to microseconds.
- static double [MillisecondsToSeconds](#) (double seconds)
Converts milliseconds to seconds.
- static uint [Min](#) (uint v, uint max)
Returns the minimum of two unsigned integer values.
- static uint [NextPowerOfTwo](#) (uint v)
Calculates the next power of two greater than or equal to a given uint value.
- static unsafe string [PrintBits](#) (byte *data, int count)
Converts a byte array to a string representation of its bits.
- static uint [QuaternionCompress](#) (Quaternion rot)
Primary Compression Method. Converts a quaternion into a ulong buffer. Depending on size most of the top bits will be 0.
- static Quaternion [QuaternionDecompress](#) (uint buffer)
Primary Decompression Method. Decompress the 3 channels and missing channel ID from the serialized ULong buffer.
- static long [SecondsToMicroseconds](#) (double seconds)
Converts seconds to microseconds.
- static long [SecondsToMilliseconds](#) (double seconds)
Converts seconds to milliseconds.
- static unsafe int [SizeOfBits< T > \(\)](#)
Returns the size of the specified unmanaged type in bits.
- static int [ZigZagDecode](#) (int i)
Decodes a ZigZag encoded integer.
- static long [ZigZagDecode](#) (long i)
Decodes a ZigZag encoded long integer.
- static int [ZigZagEncode](#) (int i)
Encodes an integer using ZigZag encoding.
- static long [ZigZagEncode](#) (long i)
Encodes a long integer using ZigZag encoding.

6.149.1 Detailed Description

Math utility methods.

6.149.2 Member Function Documentation

6.149.2.1 BitScanReverse() [1/4]

```
static int BitScanReverse (
    int v ) [static]
```

Finds the index of the most significant set bit (1-bit) in a 32-bit integer.

Parameters

<code>v</code>	The 32-bit integer to scan.
----------------	-----------------------------

Returns

The index of the most significant set bit.

6.149.2.2 BitScanReverse() [2/4]

```
static int BitScanReverse (
    long v ) [static]
```

Finds the index of the most significant set bit (1-bit) in a 64-bit integer.

Parameters

<code>v</code>	The 64-bit integer to scan.
----------------	-----------------------------

Returns

The index of the most significant set bit.

6.149.2.3 BitScanReverse() [3/4]

```
static int BitScanReverse (
    uint v ) [static]
```

Finds the index of the most significant set bit (1-bit) in a 32-bit unsigned integer.

Parameters

<code>v</code>	The 32-bit unsigned integer to scan.
----------------	--------------------------------------

Returns

The index of the most significant set bit.

6.149.2.4 BitScanReverse() [4/4]

```
static int BitScanReverse (  
    ulong v ) [static]
```

Finds the index of the most significant set bit (1-bit) in a 64-bit unsigned integer.

Parameters

<i>v</i>	The 64-bit unsigned integer to scan.
----------	--------------------------------------

Returns

The index of the most significant set bit.

6.149.2.5 BitsRequiredForNumber() [1/2]

```
static int BitsRequiredForNumber (  
    int n ) [static]
```

Determines the number of bits required to represent the specified integer.

Parameters

<i>n</i>	The integer to evaluate.
----------	--------------------------

Returns

The number of bits required to represent the integer.

6.149.2.6 BitsRequiredForNumber() [2/2]

```
static int BitsRequiredForNumber (  
    uint n ) [static]
```

Determines the number of bits required to represent a uint value.

Parameters

<i>n</i>	The uint value to evaluate.
----------	-----------------------------

Returns

The number of bits required to represent the value.

6.149.2.7 BytesRequiredForBits() [1/2]

```
static int BytesRequiredForBits (  
    int b ) [static]
```

Calculates the number of bytes required to store the specified number of bits.

Parameters

<i>b</i>	The number of bits.
----------	---------------------

Returns

The number of bytes required.

6.149.2.8 BytesRequiredForBits() [2/2]

```
static short BytesRequiredForBits (  
    short b ) [static]
```

Calculates the number of bytes required to store the specified number of bits.

Parameters

<i>b</i>	The number of bits.
----------	---------------------

Returns

The number of bytes required.

6.149.2.9 CeilToInt()

```
static int CeilToInt (  
    double value ) [static]
```

Rounds a double value up to the nearest integer.

Parameters

<i>value</i>	The double value to round up.
--------------	-------------------------------

Returns

The smallest integer greater than or equal to the specified value.

6.149.2.10 Clamp() [1/4]

```
static double Clamp (
    double v,
    double min,
    double max ) [static]
```

Clamps a double value to the specified range.

Parameters

<i>v</i>	The value to clamp.
<i>min</i>	The minimum value of the range.
<i>max</i>	The maximum value of the range.

Returns

The clamped value.

6.149.2.11 Clamp() [2/4]

```
static float Clamp (
    float v,
    float min,
    float max ) [static]
```

Clamps a float value to the specified range.

Parameters

<i>v</i>	The value to clamp.
<i>min</i>	The minimum value of the range.
<i>max</i>	The maximum value of the range.

Returns

The clamped value.

6.149.2.12 Clamp() [3/4]

```
static int Clamp (  
    int v,  
    int min,  
    int max ) [static]
```

Clamps an integer value to the specified range.

Parameters

<i>v</i>	The value to clamp.
<i>min</i>	The minimum value of the range.
<i>max</i>	The maximum value of the range.

Returns

The clamped value.

6.149.2.13 Clamp() [4/4]

```
static uint Clamp (  
    uint v,  
    uint min,  
    uint max ) [static]
```

Clamps an unsigned integer value to the specified range.

Parameters

<i>v</i>	The value to clamp.
<i>min</i>	The minimum value of the range.
<i>max</i>	The maximum value of the range.

Returns

The clamped value.

6.149.2.14 Clamp01() [1/2]

```
static double Clamp01 (  
    double v ) [static]
```

Clamps a double value to the range [0, 1].

Parameters

<i>v</i>	The value to clamp.
----------	---------------------

Returns

The clamped value.

6.149.2.15 Clamp01() [2/2]

```
static float Clamp01 (  
    float v ) [static]
```

Clamps a float value to the range [0, 1].

Parameters

<i>v</i>	The value to clamp.
----------	---------------------

Returns

The clamped value.

6.149.2.16 ClampToByte()

```
static byte ClampToByte (  
    int v ) [static]
```

Clamps an integer value to the range [0, 255] and returns it as a byte.

Parameters

<i>v</i>	The value to clamp.
----------	---------------------

Returns

The clamped value as a byte.

6.149.2.17 CosineInterpolate()

```
static double CosineInterpolate (  
    double a,  
    double b,  
    double t ) [static]
```

Performs cosine interpolation between two values.

Parameters

<i>a</i>	The start value.
<i>b</i>	The end value.
<i>t</i>	The interpolation factor, typically between 0 and 1.

Returns

The interpolated value.

6.149.2.18 CountSetBits()

```
static int CountSetBits (  
    ulong x ) [static]
```

Counts the number of bits set to 1 in a ulong value.

Parameters

<i>x</i>	The ulong value to count bits in.
----------	-----------------------------------

Returns

The number of bits set to 1.

6.149.2.19 CountUsedBitsMinOne()

```
static int CountUsedBitsMinOne (  
    uint value ) [static]
```

Counts the number of bits set to 1 in a uint value, minus one.

Parameters

<i>value</i>	The uint value to count bits in.
--------------	----------------------------------

Returns

The number of bits set to 1, minus one.

6.149.2.20 FloorToInt()

```
static int FloorToInt (  
    double value ) [static]
```

Rounds a double value down to the nearest integer.

Parameters

<i>value</i>	The double value to round down.
--------------	---------------------------------

Returns

The largest integer less than or equal to the specified value.

6.149.2.21 IntsRequiredForBits()

```
static int IntsRequiredForBits (  
    int b ) [static]
```

Calculates the number of integers required to store the specified number of bits.

Parameters

<i>b</i>	The number of bits.
----------	---------------------

Returns

The number of integers required.

6.149.2.22 Lerp() [1/2]

```
static double Lerp (  
    double a,
```



```
double b,  
double t ) [static]
```

Linearly interpolates between two double values.

Parameters

<i>a</i>	The start value.
<i>b</i>	The end value.
<i>t</i>	The interpolation factor, typically between 0 and 1.

Returns

The interpolated value.

6.149.2.23 Lerp() [2/2]

```
static float Lerp (  
    float a,  
    float b,  
    float t ) [static]
```

Linearly interpolates between two float values.

Parameters

<i>a</i>	The start value.
<i>b</i>	The end value.
<i>t</i>	The interpolation factor, typically between 0 and 1.

Returns

The interpolated value.

6.149.2.24 MicrosecondsToSeconds()

```
static double MicrosecondsToSeconds (  
    long microseconds ) [static]
```

Converts microseconds to seconds.

Parameters

<i>microseconds</i>	The time in microseconds.
---------------------	---------------------------

Returns

The time in seconds.

6.149.2.25 MillisecondsToMicroseconds()

```
static long MillisecondsToMicroseconds (  
    long milliseconds ) [static]
```

Converts milliseconds to microseconds.

Parameters

<i>milliseconds</i>	The time in milliseconds.
---------------------	---------------------------

Returns

The time in microseconds.

6.149.2.26 MillisecondsToSeconds()

```
static double MillisecondsToSeconds (  
    double seconds ) [static]
```

Converts milliseconds to seconds.

Parameters

<i>seconds</i>	The time in milliseconds.
----------------	---------------------------

Returns

The time in seconds.

6.149.2.27 Min()

```
static uint Min (  
    uint v,  
    uint max ) [static]
```

Returns the minimum of two unsigned integer values.

Parameters

<i>v</i>	The first value.
<i>max</i>	The second value.

Returns

The minimum value.

6.149.2.28 NextPowerOfTwo()

```
static uint NextPowerOfTwo (  
    uint v ) [static]
```

Calculates the next power of two greater than or equal to a given uint value.

Parameters

<i>v</i>	The uint value to evaluate.
----------	-----------------------------

Returns

The next power of two greater than or equal to the value.

6.149.2.29 PrintBits()

```
static unsafe string PrintBits (  
    byte * data,  
    int count ) [static]
```

Converts a byte array to a string representation of its bits.

Parameters

<i>data</i>	Pointer to the byte array.
<i>count</i>	The number of bytes to convert.

Returns

A string representation of the bits.

6.149.2.30 QuaternionCompress()

```
static uint QuaternionCompress (
    Quaternion rot ) [static]
```

Primary Compression Method. Converts a quaternion into a ulong buffer. Depending on size most of the top bits will be 0.

Parameters

<i>rot</i>	The quaternion to be compressed
------------	---------------------------------

Returns

A ulong buffer of the compressed quat.

6.149.2.31 QuaternionDecompress()

```
static Quaternion QuaternionDecompress (
    uint buffer ) [static]
```

Primary Decompression Method. Decompress the 3 channels and missing channel ID from the serialized ULong buffer.

Parameters

<i>buffer</i>	The ulong that represents the compressed quaternion.
---------------	--

Returns

The restored Quaternion.

6.149.2.32 SecondsToMicroseconds()

```
static long SecondsToMicroseconds (
    double seconds ) [static]
```

Converts seconds to microseconds.

Parameters

<i>seconds</i>	The time in seconds.
----------------	----------------------

Returns

The time in microseconds.

6.149.2.33 SecondsToMilliseconds()

```
static long SecondsToMilliseconds (  
    double seconds ) [static]
```

Converts seconds to milliseconds.

Parameters

<i>seconds</i>	The time in seconds.
----------------	----------------------

Returns

The time in milliseconds.

6.149.2.34 SizeOfBits< T >()

```
static unsafe int SizeOfBits< T > ( ) [static]
```

Returns the size of the specified unmanaged type in bits.

Template Parameters

<i>T</i>	The unmanaged type.
----------	---------------------

Returns

The size of the type in bits.

Type Constraints

T* : *unmanaged

6.149.2.35 ZigZagDecode() [1/2]

```
static int ZigZagDecode (  
    int i ) [static]
```

Decodes a ZigZag encoded integer.

Parameters

<i>i</i>	The ZigZag encoded integer to decode.
----------	---------------------------------------

Returns

The decoded integer.

6.149.2.36 ZigZagDecode() [2/2]

```
static long ZigZagDecode (  
    long i ) [static]
```

Decodes a ZigZag encoded long integer.

Parameters

<i>i</i>	The ZigZag encoded long integer to decode.
----------	--

Returns

The decoded long integer.

6.149.2.37 ZigZagEncode() [1/2]

```
static int ZigZagEncode (  
    int i ) [static]
```

Encodes an integer using ZigZag encoding.

Parameters

<i>i</i>	The integer to encode.
----------	------------------------

Returns

The ZigZag encoded integer.

6.149.2.38 ZigZagEncode() [2/2]

```
static long ZigZagEncode (  
    long i ) [static]
```

Encodes a long integer using ZigZag encoding.

Parameters

<i>i</i>	The long integer to encode.
----------	-----------------------------

Returns

The ZigZag encoded long integer.

6.150 Maths.FastAbs Struct Reference

Represents a structure that allows quick setting of the sign bit of floats using field offsets.

Public Attributes

- float [Single](#)
The single-precision floating-point number.
- uint [UInt32](#)
The unsigned integer representation of the float.

Static Public Attributes

- const uint [Mask](#) = 0x7FFFFFFF
The mask used to clear the sign bit of a float.

6.150.1 Detailed Description

Represents a structure that allows quick setting of the sign bit of floats using field offsets.

6.150.2 Member Data Documentation

6.150.2.1 Mask

```
const uint Mask = 0x7FFFFFFF [static]
```

The mask used to clear the sign bit of a float.

6.150.2.2 Single

```
float Single
```

The single-precision floating-point number.

6.150.2.3 UInt32

```
uint UInt32
```

The unsigned integer representation of the float.

6.151 MaxStringByteCountAttribute Class Reference

Specifies that the string field should be drawn as a text field with a maximum byte count for given encoding.

Inherits [DrawerPropertyAttribute](#).

Public Member Functions

- [MaxStringByteCountAttribute](#) (int count, string encoding)

Initializes a new instance of the [MaxStringByteCountAttribute](#) class with the specified byte count and encoding.

Properties

- int [ByteCount](#) [get]
Maximum byte count for the string.
- string [Encoding](#) [get]
The encoding of the string.

6.151.1 Detailed Description

Specifies that the string field should be drawn as a text field with a maximum byte count for given encoding.

6.151.2 Constructor & Destructor Documentation

6.151.2.1 MaxStringByteCountAttribute()

```
MaxStringByteCountAttribute (  
    int count,  
    string encoding )
```

Initializes a new instance of the [MaxStringByteCountAttribute](#) class with the specified byte count and encoding.

Parameters

<i>count</i>	
<i>encoding</i>	

6.151.3 Property Documentation

6.151.3.1 ByteCount

`int ByteCount [get]`

Maximum byte count for the string.

6.151.3.2 Encoding

`string Encoding [get]`

The encoding of the string.

6.152 Native Class Reference

[Native Memory Allocator](#)

Static Public Member Functions

- static void * [AlignPointer](#) (void *pointer, int alignment)
Aligns a pointer to a specified alignment.
- static void [ArrayClear](#)< T > (T *ptr, int size)
Clears a range of elements in an array by setting them to zero.
- static int [ArrayCompare](#)< T > (T *ptr1, T *ptr2, int size)
Compares two arrays for equality.
- static void [ArrayCopy](#) (void *source, int sourceIndex, void *destination, int destinationIndex, int count, int elementStride)
Copies a range of elements from one array to another.
- static int [CopyFromArray](#)< T > (void *destination, T[] source)
Copies elements from a managed array to an unmanaged memory location.
- static int [CopyToArray](#)< T > (T[] destination, void *source)
Copies elements from an unmanaged memory location to a managed array.
- static T * [DoubleArray](#)< T > (T *array, int currentLength)
Doubles the size of an array by expanding it to twice its current length.
- static T ** [DoublePtrArray](#)< T > (T **array, int currentLength)
Doubles the size of an array of pointers by expanding it to twice its current length.
- static T [Empty](#)< T > ()
Returns an empty instance of the specified unmanaged type.
- static void * [Expand](#) (void *buffer, int currentSize, int newSize)
Expands a buffer to a new specified size.
- static T * [ExpandArray](#)< T > (T *array, int currentLength, int newLength)
Expands an array to a new specified length.

- static T ** [ExpandPtrArray](#)< T > (T **array, int currentLength, int newLength)
Expands an array of pointers to a new specified length.
- static void [Free](#) (void *memory)
Frees a block of memory.
- static int [GetAlignment](#) (int stride)
Gets the alignment of the specified stride.
- static int [GetAlignment](#)< T > ()
Gets the alignment of the specified unmanaged type.
- static int [GetFieldOffset](#) (System.Reflection.FieldInfo fi)
Gets the offset of the specified field.
- static int [GetLengthPrefixedUTF8ByteCount](#) (string str)
Gets the byte count of a UTF-8 encoded string with a length prefix.
- static int [GetMaxAlignment](#) (int a, int b)
Gets the maximum alignment of two given strides.
- static int [GetMaxAlignment](#) (int a, int b, int c)
Gets the maximum alignment of three given strides.
- static int [GetMaxAlignment](#) (int a, int b, int c, int d)
Gets the maximum alignment of four given strides.
- static int [GetMaxAlignment](#) (int a, int b, int c, int d, int e)
Gets the maximum alignment of five given strides.
- static bool [IsAligned](#) (int stride, int alignment)
Checks if a stride is aligned to a specified alignment.
- static bool [IsPointerAligned](#) (void *pointer, int alignment)
Checks if a pointer is aligned to a specified alignment.
- static void * [Malloc](#) (int size)
Allocates a block of memory.
- static T * [Malloc](#)< T > ()
Allocates a block of memory for a specified type.
- static void * [MallocAndClear](#) (int size)
Allocates and clears a block of memory.
- static T * [MallocAndClear](#)< T > ()
Allocates and clears a block of memory for a specified type.
- static void * [MallocAndClearArray](#) (int stride, int length)
Allocates and clears an array of memory blocks.
- static T * [MallocAndClearArray](#)< T > (int length)
Allocates and clears an array of memory blocks for a specified type.
- static T * [MallocAndClearArrayMin1](#)< T > (int length)
Allocates and clears an array of memory blocks for a specified type, ensuring a minimum length of 1.
- static int [MallocAndClearBlock](#) (int size0, int size1, int size2, int size3, int size4, int size5, int size6, int size7, int size8, int size9, int size10, int size11, out void *ptr0, out void *ptr1, out void *ptr2, out void *ptr3, out void *ptr4, out void *ptr5, out void *ptr6, out void *ptr7, out void *ptr8, out void *ptr9, out void *ptr10, out void *ptr11, int alignment=[ALIGNMENT](#))
Allocates a block of memory and clears it.
- static int [MallocAndClearBlock](#) (int size0, int size1, int size2, int size3, int size4, int size5, int size6, int size7, int size8, int size9, int size10, out void *ptr0, out void *ptr1, out void *ptr2, out void *ptr3, out void *ptr4, out void *ptr5, out void *ptr6, out void *ptr7, out void *ptr8, out void *ptr9, out void *ptr10, int alignment=[ALIGNMENT](#))
Allocates a block of memory and clears it.
- static int [MallocAndClearBlock](#) (int size0, int size1, int size2, int size3, int size4, int size5, int size6, int size7, int size8, int size9, out void *ptr0, out void *ptr1, out void *ptr2, out void *ptr3, out void *ptr4, out void *ptr5, out void *ptr6, out void *ptr7, out void *ptr8, out void *ptr9, int alignment=[ALIGNMENT](#))
Allocates a block of memory and clears it.

- static int [MallocAndClearBlock](#) (int size0, int size1, int size2, int size3, int size4, int size5, int size6, int size7, int size8, out void *ptr0, out void *ptr1, out void *ptr2, out void *ptr3, out void *ptr4, out void *ptr5, out void *ptr6, out void *ptr7, out void *ptr8, int alignment=[ALIGNMENT](#))
Allocates a block of memory and clears it.
- static int [MallocAndClearBlock](#) (int size0, int size1, int size2, int size3, int size4, int size5, int size6, int size7, out void *ptr0, out void *ptr1, out void *ptr2, out void *ptr3, out void *ptr4, out void *ptr5, out void *ptr6, out void *ptr7, int alignment=[ALIGNMENT](#))
Allocates a block of memory and clears it.
- static int [MallocAndClearBlock](#) (int size0, int size1, int size2, int size3, int size4, int size5, int size6, out void *ptr0, out void *ptr1, out void *ptr2, out void *ptr3, out void *ptr4, out void *ptr5, out void *ptr6, int alignment=[ALIGNMENT](#))
Allocates a block of memory and clears it.
- static int [MallocAndClearBlock](#) (int size0, int size1, int size2, int size3, int size4, int size5, out void *ptr0, out void *ptr1, out void *ptr2, out void *ptr3, out void *ptr4, out void *ptr5, int alignment=[ALIGNMENT](#))
Allocates a block of memory and clears it.
- static int [MallocAndClearBlock](#) (int size0, int size1, int size2, int size3, int size4, out void *ptr0, out void *ptr1, out void *ptr2, out void *ptr3, out void *ptr4, int alignment=[ALIGNMENT](#))
Allocates a block of memory and clears it.
- static int [MallocAndClearBlock](#) (int size0, int size1, int size2, int size3, out void *ptr0, out void *ptr1, out void *ptr2, out void *ptr3, int alignment=[ALIGNMENT](#))
Allocates a block of memory and clears it.
- static int [MallocAndClearBlock](#) (int size0, int size1, int size2, out void *ptr0, out void *ptr1, out void *ptr2, int alignment=[ALIGNMENT](#))
Allocates a block of memory and clears it.
- static int [MallocAndClearBlock](#) (int size0, int size1, out void *ptr0, out void *ptr1, int alignment=[ALIGNMENT](#))
Allocates a block of memory and clears it.
- static T ** [MallocAndClearPtrArray](#)< T > (int length)
Allocates and clears an array of pointers for a specified type.
- static T ** [MallocAndClearPtrArrayMin1](#)< T > (int length)
Allocates and clears an array of pointers for a specified type, ensuring a minimum length of 1.
- static void [MemClear](#) (void *ptr, int size)
Clears a block of memory by setting it to zero.
- static int [MemCmp](#) (void *ptr1, void *ptr2, int size)
Compares two blocks of memory.
- static void [MemCpy](#) (void *destination, void *source, int size)
Copies a block of memory from the source to the destination.
- static void [MemCpyFast](#) (void *d, void *s, int size)
Copies memory quickly for specific sizes.
- static void [MemMove](#) (void *destination, void *source, int size)
Moves a block of memory from the source to the destination.
- static void [PrintAllocatedButNotFreed](#) ()
- static int [ReadLengthPrefixedUTF8](#) (void *source, out string result)
Reads a length-prefixed UTF-8 encoded string from a memory location.
- static unsafe byte * [ReferenceToPointer](#)< T > (ref T obj)
Converts a reference to an unmanaged type to a pointer.
- static int [RoundBitsUpTo32](#) (int bits)
Rounds the given number of bits up to the nearest multiple of 32.
- static int [RoundBitsUpTo64](#) (int bits)
Rounds the given number of bits up to the nearest multiple of 64.
- static int [RoundToAlignment](#) (int stride, int alignment)
Rounds a stride to a specified alignment.
- static long [RoundToAlignment](#) (long stride, int alignment)

- Rounds a stride to a specified alignment.*

 - static int [RoundToMaxAlignment](#) (int stride)
- Rounds a stride to the maximum alignment.*

 - static int [SizeOf](#) (Type t)
- Gets the size of the specified type.*

 - static int [WordCount](#) (int stride, int wordSize)
- Calculates the word count for a given stride and word size.*

 - static int [WriteLengthPrefixedUTF8](#) (void *destination, string str)
- Writes a length-prefixed UTF-8 encoded string to a memory location.*

Static Public Attributes

- const int [ALIGNMENT](#) = 8
 - The alignment size in bytes.*
- const int [CACHE_LINE_SIZE](#) = 64
 - The size of the cache line in bytes.*

6.152.1 Detailed Description

[Native Memory Allocator](#)

6.152.2 Member Function Documentation

6.152.2.1 [AlignPointer\(\)](#)

```
static void* AlignPointer (
    void * pointer,
    int alignment ) [static]
```

Aligns a pointer to a specified alignment.

Parameters

<i>pointer</i>	The pointer to align.
<i>alignment</i>	The alignment to apply.

Returns

The aligned pointer.

6.152.2.2 ArrayClear< T >()

```
static void ArrayClear< T > (
    T * ptr,
    int size ) [static]
```

Clears a range of elements in an array by setting them to zero.

Template Parameters

<i>T</i>	The type of the elements in the array.
----------	--

Parameters

<i>ptr</i>	The pointer to the array.
<i>size</i>	The number of elements to clear.

Type Constraints

T* : *unmanaged

6.152.2.3 ArrayCompare< T >()

```
static int ArrayCompare< T > (
    T * ptr1,
    T * ptr2,
    int size ) [static]
```

Compares two arrays for equality.

Template Parameters

<i>T</i>	The type of the elements in the arrays.
----------	---

Parameters

<i>ptr1</i>	The pointer to the first array.
<i>ptr2</i>	The pointer to the second array.
<i>size</i>	The number of elements to compare.

Returns

An integer that indicates the relative order of the arrays being compared.

Type Constraints

T* : *unmanaged

6.152.2.4 ArrayCopy()

```
static void ArrayCopy (
    void * source,
    int sourceIndex,
    void * destination,
    int destinationIndex,
    int count,
    int elementStride ) [static]
```

Copies a range of elements from one array to another.

Parameters

<i>source</i>	The source array pointer.
<i>sourceIndex</i>	The zero-based index in the source array at which copying begins.
<i>destination</i>	The destination array pointer.
<i>destinationIndex</i>	The zero-based index in the destination array at which storing begins.
<i>count</i>	The number of elements to copy.
<i>elementStride</i>	The size of each element in bytes.

6.152.2.5 CopyFromArray< T >()

```
static int CopyFromArray< T > (
    void * destination,
    T[] source ) [static]
```

Copies elements from a managed array to an unmanaged memory location.

Template Parameters

<i>T</i>	The type of the elements in the array.
----------	--

Parameters

<i>destination</i>	The destination pointer.
<i>source</i>	The source array.

Returns

The number of bytes copied.

Type Constraints

T: *unmanaged*

6.152.2.6 CopyToArray< T >()

```
static int CopyToArray< T > (
    T[] destination,
    void * source ) [static]
```

Copies elements from an unmanaged memory location to a managed array.

Template Parameters

<i>T</i>	The type of the elements in the array.
----------	--

Parameters

<i>destination</i>	The destination array.
<i>source</i>	The source pointer.

Returns

The number of bytes copied.

Type Constraints

T : unmanaged

6.152.2.7 DoubleArray< T >()

```
static T* DoubleArray< T > (
    T * array,
    int currentLength ) [static]
```

Doubles the size of an array by expanding it to twice its current length.

Template Parameters

<i>T</i>	The type of the elements in the array.
----------	--

Parameters

<i>array</i>	The pointer to the array to double.
<i>currentLength</i>	The current length of the array.

Returns

A pointer to the new array with doubled size.

Type Constraints

T : unmanaged

6.152.2.8 DoublePtrArray< T >()

```
static T** DoublePtrArray< T > (
    T ** array,
    int currentLength ) [static]
```

Doubles the size of an array of pointers by expanding it to twice its current length.

Template Parameters

<i>T</i>	The type of the elements in the array.
----------	--

Parameters

<i>array</i>	The pointer to the array of pointers to double.
<i>currentLength</i>	The current length of the array.

Returns

A pointer to the new array of pointers with doubled size.

Type Constraints

T* : *unmanaged

6.152.2.9 Empty< T >()

```
static T Empty< T > ( ) [static]
```

Returns an empty instance of the specified unmanaged type.

Template Parameters

<i>T</i>	The type of the instance to return.
----------	-------------------------------------

Returns

An empty instance of the specified type.

Type Constraints

T* : *unmanaged

6.152.2.10 Expand()

```
static void* Expand (
    void * buffer,
    int currentSize,
    int newSize ) [static]
```

Expands a buffer to a new specified size.

Parameters

<i>buffer</i>	The pointer to the buffer to expand.
<i>currentSize</i>	The current size of the buffer.
<i>newSize</i>	The new size of the buffer.

Returns

A pointer to the new expanded buffer.

6.152.2.11 ExpandArray< T >()

```
static T* ExpandArray< T > (
    T * array,
    int currentLength,
    int newLength ) [static]
```

Expands an array to a new specified length.

Template Parameters

<i>T</i>	The type of the elements in the array.
----------	--

Parameters

<i>array</i>	The pointer to the array to expand.
<i>currentLength</i>	The current length of the array.
<i>newLength</i>	The new length of the array.

Returns

A pointer to the new expanded array.

Type Constraints

T* : *unmanaged

6.152.2.12 ExpandPtrArray< T >()

```
static T** ExpandPtrArray< T > (
    T ** array,
    int currentLength,
    int newLength ) [static]
```

Expands an array of pointers to a new specified length.

Template Parameters

<i>T</i>	The type of the elements in the array.
----------	--

Parameters

<i>array</i>	The pointer to the array of pointers to expand.
<i>currentLength</i>	The current length of the array.
<i>newLength</i>	The new length of the array.

Returns

A pointer to the new expanded array of pointers.

Type Constraints

T*: *unmanaged

6.152.2.13 Free()

```
static void Free (
    void * memory ) [static]
```

Frees a block of memory.

Parameters

<i>memory</i>	The pointer to the memory block to free.
---------------	--

6.152.2.14 GetAlignment()

```
static int GetAlignment (
    int stride ) [static]
```

Gets the alignment of the specified stride.

Parameters

<i>stride</i>	The stride to get the alignment of.
---------------	-------------------------------------

Returns

The alignment of the specified stride.

6.152.2.15 GetAlignment< T >()

```
static int GetAlignment< T > ( ) [static]
```

Gets the alignment of the specified unmanaged type.

Template Parameters

<i>T</i>	The type to get the alignment of.
----------	-----------------------------------

Returns

The alignment of the specified type.

Type Constraints

T*: *unmanaged

6.152.2.16 GetFieldOffset()

```
static int GetFieldOffset (
    System.Reflection.FieldInfo fi ) [static]
```

Gets the offset of the specified field.

Parameters

<i>fi</i>	The field information.
-----------	------------------------

Returns

The offset of the specified field in bytes.

6.152.2.17 GetLengthPrefixedUTF8ByteCount()

```
static int GetLengthPrefixedUTF8ByteCount (  
    string str ) [static]
```

Gets the byte count of a UTF-8 encoded string with a length prefix.

Parameters

<i>str</i>	The string to encode.
------------	-----------------------

Returns

The byte count of the encoded string with the length prefix.

6.152.2.18 GetMaxAlignment() [1/4]

```
static int GetMaxAlignment (  
    int a,  
    int b ) [static]
```

Gets the maximum alignment of two given strides.

Parameters

<i>a</i>	The first stride.
<i>b</i>	The second stride.

Returns

The maximum alignment of the two strides.

6.152.2.19 GetMaxAlignment() [2/4]

```
static int GetMaxAlignment (  
    int a,  
    int b,  
    int c ) [static]
```

Gets the maximum alignment of three given strides.

Parameters

<i>a</i>	The first stride.
<i>b</i>	The second stride.
<i>c</i>	The third stride.

Returns

The maximum alignment of the three strides.

6.152.2.20 GetMaxAlignment() [3/4]

```
static int GetMaxAlignment (  
    int a,  
    int b,  
    int c,  
    int d ) [static]
```

Gets the maximum alignment of four given strides.

Parameters

<i>a</i>	The first stride.
<i>b</i>	The second stride.
<i>c</i>	The third stride.
<i>d</i>	The fourth stride.

Returns

The maximum alignment of the four strides.

6.152.2.21 GetMaxAlignment() [4/4]

```
static int GetMaxAlignment (  
    int a,  
    int b,  
    int c,  
    int d,  
    int e ) [static]
```

Gets the maximum alignment of five given strides.

Parameters

<i>a</i>	The first stride.
<i>b</i>	The second stride.
<i>c</i>	The third stride.
<i>d</i>	The fourth stride.
<i>e</i>	The fifth stride.

Returns

The maximum alignment of the five strides.

6.152.2.22 IsAligned()

```
static bool IsAligned (
    int stride,
    int alignment ) [static]
```

Checks if a stride is aligned to a specified alignment.

Parameters

<i>stride</i>	The stride to check.
<i>alignment</i>	The alignment to check against.

Returns

True if the stride is aligned, otherwise false.

6.152.2.23 IsPointerAligned()

```
static bool IsPointerAligned (
    void * pointer,
    int alignment ) [static]
```

Checks if a pointer is aligned to a specified alignment.

Parameters

<i>pointer</i>	The pointer to check.
<i>alignment</i>	The alignment to check against.

Returns

True if the pointer is aligned, otherwise false.

6.152.2.24 Malloc()

```
static void* Malloc (
    int size ) [static]
```

Allocates a block of memory.

Parameters

<i>size</i>	The number of bytes to allocate.
-------------	----------------------------------

Returns

A pointer to the allocated memory block.

Exceptions

<i>Exception</i>	Thrown when the size is less than or equal to zero or exceeds the maximum allowed size.
------------------	---

6.152.2.25 Malloc< T >()

```
static T* Malloc< T > ( ) [static]
```

Allocates a block of memory for a specified type.

Template Parameters

<i>T</i>	The type of the memory block to allocate.
----------	---

Returns

A pointer to the allocated memory block.

Type Constraints

T* : *unmanaged

6.152.2.26 MallocAndClear()

```
static void* MallocAndClear (
    int size ) [static]
```

Allocates and clears a block of memory.

Parameters

<i>size</i>	The number of bytes to allocate and clear.
-------------	--

Returns

A pointer to the allocated and cleared memory block.

6.152.2.27 MallocAndClear< T >()

```
static T* MallocAndClear< T > ( ) [static]
```

Allocates and clears a block of memory for a specified type.

Template Parameters

<i>T</i>	The type of the memory block to allocate and clear.
----------	---

Returns

A pointer to the allocated and cleared memory block.

Type Constraints

T: *unmanaged*

6.152.2.28 MallocAndClearArray()

```
static void* MallocAndClearArray (
    int stride,
    int length ) [static]
```

Allocates and clears an array of memory blocks.

Parameters

<i>stride</i>	The size of each element in the array.
<i>length</i>	The number of elements in the array.

Returns

A pointer to the allocated and cleared array of memory blocks.

6.152.2.29 MallocAndClearArray< T >()

```
static T* MallocAndClearArray< T > (
    int length ) [static]
```

Allocates and clears an array of memory blocks for a specified type.

Template Parameters

<i>T</i>	The type of the elements in the array.
----------	--

Parameters

<i>length</i>	The number of elements in the array.
---------------	--------------------------------------

Returns

A pointer to the allocated and cleared array of memory blocks.

Type Constraints

T*: *unmanaged

6.152.2.30 MallocAndClearArrayMin1< T >()

```
static T* MallocAndClearArrayMin1< T > (  
    int length ) [static]
```

Allocates and clears an array of memory blocks for a specified type, ensuring a minimum length of 1.

Template Parameters

<i>T</i>	The type of the elements in the array.
----------	--

Parameters

<i>length</i>	The number of elements in the array.
---------------	--------------------------------------

Returns

A pointer to the allocated and cleared array of memory blocks.

Type Constraints

T*: *unmanaged

6.152.2.31 MallocAndClearBlock() [1/11]

```
static int MallocAndClearBlock (  
    int size0,
```

```
int size1,  
int size2,  
int size3,  
int size4,  
int size5,  
int size6,  
int size7,  
int size8,  
int size9,  
int size10,  
int size11,  
out void * ptr0,  
out void * ptr1,  
out void * ptr2,  
out void * ptr3,  
out void * ptr4,  
out void * ptr5,  
out void * ptr6,  
out void * ptr7,  
out void * ptr8,  
out void * ptr9,  
out void * ptr10,  
out void * ptr11,  
int alignment = ALIGNMENT ) [static]
```

Allocates a block of memory and clears it.

6.152.2.32 MallocAndClearBlock() [2/11]

```
static int MallocAndClearBlock (  
int size0,  
int size1,  
int size2,  
int size3,  
int size4,  
int size5,  
int size6,  
int size7,  
int size8,  
int size9,  
int size10,  
out void * ptr0,  
out void * ptr1,  
out void * ptr2,  
out void * ptr3,  
out void * ptr4,  
out void * ptr5,  
out void * ptr6,  
out void * ptr7,  
out void * ptr8,  
out void * ptr9,  
out void * ptr10,  
int alignment = ALIGNMENT ) [static]
```

Allocates a block of memory and clears it.

6.152.2.33 MallocAndClearBlock() [3/11]

```
static int MallocAndClearBlock (
    int size0,
    int size1,
    int size2,
    int size3,
    int size4,
    int size5,
    int size6,
    int size7,
    int size8,
    int size9,
    out void * ptr0,
    out void * ptr1,
    out void * ptr2,
    out void * ptr3,
    out void * ptr4,
    out void * ptr5,
    out void * ptr6,
    out void * ptr7,
    out void * ptr8,
    out void * ptr9,
    int alignment = ALIGNMENT ) [static]
```

Allocates a block of memory and clears it.

6.152.2.34 MallocAndClearBlock() [4/11]

```
static int MallocAndClearBlock (
    int size0,
    int size1,
    int size2,
    int size3,
    int size4,
    int size5,
    int size6,
    int size7,
    int size8,
    out void * ptr0,
    out void * ptr1,
    out void * ptr2,
    out void * ptr3,
    out void * ptr4,
    out void * ptr5,
    out void * ptr6,
    out void * ptr7,
    out void * ptr8,
    int alignment = ALIGNMENT ) [static]
```

Allocates a block of memory and clears it.

6.152.2.35 MallocAndClearBlock() [5/11]

```
static int MallocAndClearBlock (
    int size0,
    int size1,
    int size2,
    int size3,
    int size4,
    int size5,
    int size6,
    int size7,
    out void * ptr0,
    out void * ptr1,
    out void * ptr2,
    out void * ptr3,
    out void * ptr4,
    out void * ptr5,
    out void * ptr6,
    out void * ptr7,
    int alignment = ALIGNMENT ) [static]
```

Allocates a block of memory and clears it.

6.152.2.36 MallocAndClearBlock() [6/11]

```
static int MallocAndClearBlock (
    int size0,
    int size1,
    int size2,
    int size3,
    int size4,
    int size5,
    int size6,
    out void * ptr0,
    out void * ptr1,
    out void * ptr2,
    out void * ptr3,
    out void * ptr4,
    out void * ptr5,
    out void * ptr6,
    int alignment = ALIGNMENT ) [static]
```

Allocates a block of memory and clears it.

6.152.2.37 MallocAndClearBlock() [7/11]

```
static int MallocAndClearBlock (
    int size0,
    int size1,
    int size2,
```

```
int size3,  
int size4,  
int size5,  
out void * ptr0,  
out void * ptr1,  
out void * ptr2,  
out void * ptr3,  
out void * ptr4,  
out void * ptr5,  
int alignment = ALIGNMENT ) [static]
```

Allocates a block of memory and clears it.

6.152.2.38 MallocAndClearBlock() [8/11]

```
static int MallocAndClearBlock (  
    int size0,  
    int size1,  
    int size2,  
    int size3,  
    int size4,  
    out void * ptr0,  
    out void * ptr1,  
    out void * ptr2,  
    out void * ptr3,  
    out void * ptr4,  
    int alignment = ALIGNMENT ) [static]
```

Allocates a block of memory and clears it.

6.152.2.39 MallocAndClearBlock() [9/11]

```
static int MallocAndClearBlock (  
    int size0,  
    int size1,  
    int size2,  
    int size3,  
    out void * ptr0,  
    out void * ptr1,  
    out void * ptr2,  
    out void * ptr3,  
    int alignment = ALIGNMENT ) [static]
```

Allocates a block of memory and clears it.

6.152.2.40 MallocAndClearBlock() [10/11]

```
static int MallocAndClearBlock (
    int size0,
    int size1,
    int size2,
    out void * ptr0,
    out void * ptr1,
    out void * ptr2,
    int alignment = ALIGNMENT ) [static]
```

Allocates a block of memory and clears it.

6.152.2.41 MallocAndClearBlock() [11/11]

```
static int MallocAndClearBlock (
    int size0,
    int size1,
    out void * ptr0,
    out void * ptr1,
    int alignment = ALIGNMENT ) [static]
```

Allocates a block of memory and clears it.

6.152.2.42 MallocAndClearPtrArray< T >()

```
static T** MallocAndClearPtrArray< T > (
    int length ) [static]
```

Allocates and clears an array of pointers for a specified type.

Template Parameters

<i>T</i>	The type of the elements in the array.
----------	--

Parameters

<i>length</i>	The number of elements in the array.
---------------	--------------------------------------

Returns

A pointer to the allocated and cleared array of pointers.

Type Constraints

T: *unmanaged*

6.152.2.43 MallocAndClearPtrArrayMin1< T >()

```
static T** MallocAndClearPtrArrayMin1< T > (
    int length ) [static]
```

Allocates and clears an array of pointers for a specified type, ensuring a minimum length of 1.

Template Parameters

<i>T</i>	The type of the elements in the array.
----------	--

Parameters

<i>length</i>	The number of elements in the array.
---------------	--------------------------------------

Returns

A pointer to the allocated and cleared array of pointers.

Type Constraints

T* : *unmanaged

6.152.2.44 MemClear()

```
static void MemClear (
    void * ptr,
    int size ) [static]
```

Clears a block of memory by setting it to zero.

Parameters

<i>ptr</i>	The pointer to the memory block.
<i>size</i>	The number of bytes to clear.

6.152.2.45 MemCmp()

```
static int MemCmp (
    void * ptr1,
    void * ptr2,
    int size ) [static]
```

Compares two blocks of memory.

Parameters

<i>ptr1</i>	The first memory block pointer.
<i>ptr2</i>	The second memory block pointer.
<i>size</i>	The number of bytes to compare.

Returns

An integer that indicates the relative order of the memory blocks being compared.

6.152.2.46 MemCpy()

```
static void MemCpy (  
    void * destination,  
    void * source,  
    int size ) [static]
```

Copies a block of memory from the source to the destination.

Parameters

<i>destination</i>	The destination pointer.
<i>source</i>	The source pointer.
<i>size</i>	The number of bytes to copy.

6.152.2.47 MemCpyFast()

```
static void MemCpyFast (  
    void * d,  
    void * s,  
    int size ) [static]
```

Copies memory quickly for specific sizes.

Parameters

<i>d</i>	The destination pointer.
<i>s</i>	The source pointer.
<i>size</i>	The number of bytes to copy.

6.152.2.48 MemMove()

```
static void MemMove (
    void * destination,
    void * source,
    int size ) [static]
```

Moves a block of memory from the source to the destination.

Parameters

<i>destination</i>	The destination pointer.
<i>source</i>	The source pointer.
<i>size</i>	The number of bytes to move.

6.152.2.49 ReadLengthPrefixedUTF8()

```
static int ReadLengthPrefixedUTF8 (
    void * source,
    out string result ) [static]
```

Reads a length-prefixed UTF-8 encoded string from a memory location.

Parameters

<i>source</i>	The source pointer.
<i>result</i>	The resulting string.

Returns

The number of bytes read.

6.152.2.50 ReferenceToPointer< T >()

```
static unsafe byte* ReferenceToPointer< T > (
    ref T obj ) [static]
```

Converts a reference to an unmanaged type to a pointer.

Template Parameters

<i>T</i>	The type of the object.
----------	-------------------------

Parameters

<i>obj</i>	The reference to the object.
------------	------------------------------

Returns

A pointer to the object.

Type Constraints

T* : *unmanaged

6.152.2.51 RoundBitsUpTo32()

```
static int RoundBitsUpTo32 (  
    int bits ) [static]
```

Rounds the given number of bits up to the nearest multiple of 32.

Parameters

<i>bits</i>	The number of bits to round up.
-------------	---------------------------------

Returns

The rounded number of bits.

6.152.2.52 RoundBitsUpTo64()

```
static int RoundBitsUpTo64 (  
    int bits ) [static]
```

Rounds the given number of bits up to the nearest multiple of 64.

Parameters

<i>bits</i>	The number of bits to round up.
-------------	---------------------------------

Returns

The rounded number of bits.

6.152.2.53 RoundToAlignment() [1/2]

```
static int RoundToAlignment (
    int stride,
    int alignment ) [static]
```

Rounds a stride to a specified alignment.

Parameters

<i>stride</i>	The stride to round.
<i>alignment</i>	The alignment to apply.

Returns

The rounded stride.

Exceptions

<i>InvalidOperationException</i>	Thrown when the alignment is invalid.
----------------------------------	---------------------------------------

6.152.2.54 RoundToAlignment() [2/2]

```
static long RoundToAlignment (
    long stride,
    int alignment ) [static]
```

Rounds a stride to a specified alignment.

Parameters

<i>stride</i>	The stride to round.
<i>alignment</i>	The alignment to apply.

Returns

The rounded stride.

Exceptions

<i>InvalidOperationException</i>	Thrown when the alignment is invalid.
----------------------------------	---------------------------------------

6.152.2.55 RoundToMaxAlignment()

```
static int RoundToMaxAlignment (
    int stride ) [static]
```

Rounds a stride to the maximum alignment.

Parameters

<i>stride</i>	The stride to round.
---------------	----------------------

Returns

The rounded stride.

6.152.2.56 SizeOf()

```
static int SizeOf (
    Type t ) [static]
```

Gets the size of the specified type.

Parameters

<i>t</i>	The type to get the size of.
----------	------------------------------

Returns

The size of the specified type in bytes.

6.152.2.57 WordCount()

```
static int WordCount (
    int stride,
    int wordSize ) [static]
```

Calculates the word count for a given stride and word size.

Parameters

<i>stride</i>	The stride to calculate.
<i>wordSize</i>	The size of each word.

Returns

The word count.

6.152.2.58 WriteLengthPrefixedUTF8()

```
static int WriteLengthPrefixedUTF8 (
    void * destination,
    string str ) [static]
```

Writes a length-prefixed UTF-8 encoded string to a memory location.

Parameters

<i>destination</i>	The destination pointer.
<i>str</i>	The string to encode and write.

Returns

The number of bytes written.

6.152.3 Member Data Documentation**6.152.3.1 ALIGNMENT**

```
const int ALIGNMENT = 8 [static]
```

The alignment size in bytes.

6.152.3.2 CACHE_LINE_SIZE

```
const int CACHE_LINE_SIZE = 64 [static]
```

The size of the cache line in bytes.

6.153 NestedComponentUtilities Class Reference

Tools to replace GetComponent variants that respects nested objects. These are used to find components of a NetworkedObjects without also finding components that belong to parent or child NetworkedObjects.

Static Public Member Functions

- static T [EnsureRootComponentExists< T, TStopOn >](#) (this Transform transform)

Ensures that a component of type T exists on the root object of the provided transform. If a component of type T does not exist, it is added. The search for the root object stops if a component of type TStopOn is found.
- static T[] [FindObjectsOfTypeInOrder< T >](#) (this UnityEngine.SceneManagement.Scene scene, bool includeInactive=false)

Find All instances of Component type in a scene. Attempts to respect the hierarchy of the scene objects to produce a more deterministic order. This is a slower operation, and does produce garbage collection.
- static void [FindObjectsOfTypeInOrder< T >](#) (this UnityEngine.SceneManagement.Scene scene, List< T > list, bool includeInactive=false)

Find All instances of Component type in a scene. Attempts to respect the hierarchy of the scene objects to produce a more deterministic order. This is a slower operation which should not be run every update.
- static TCast[] [FindObjectsOfTypeInOrder< T, TCast >](#) (this UnityEngine.SceneManagement.Scene scene, bool includeInactive=false)

Find All instances of Component type in a scene. Attempts to respect the hierarchy of the scene objects to produce a more deterministic order. This is a slow operation, and does produce garbage collection.
- static void [FindObjectsOfTypeInOrder< T, TCast >](#) (this UnityEngine.SceneManagement.Scene scene, List< TCast > list, bool includeInactive=false)

Find All instances of Component type in a scene. Attempts to respect the hierarchy of the scene objects to produce a more deterministic order. This is a slower operation and should not be run every update.
- static T [GetNestedComponentInChildren< T, TStopOn >](#) (this Transform t, bool includeInactive)

Finds the first component of type T in the children of the provided transform, stopping the search if a component of type TStopOn is found.
- static T [GetNestedComponentInParent< T, TStopOn >](#) (this Transform t)

Same as GetComponentInParent, but will always include inactive objects in search. Will also stop recursing up the hierarchy when the StopOnT is found.
- static T [GetNestedComponentInParents< T, TStopOn >](#) (this Transform t)

Finds the first component of type T in the parents of the provided transform, stopping the search if a component of type TStopOn is found.
- static List< T > [GetNestedComponentsInChildren< T >](#) (this Transform t, List< T > list, bool includeInactive=true, params System.Type[] stopOn)

Same as GetComponentInChildren, but will not recurse into children with any component of the types in the stopOn array.
- static void [GetNestedComponentsInChildren< T, TSearch, TStop >](#) (this Transform t, bool includeInactive, List< T > list)

Same as GetComponentInChildren, but will not recurse into children with component of the StopT type.
- static List< T > [GetNestedComponentsInChildren< T, TStopOn >](#) (this Transform t, List< T > list, bool includeInactive=true)

Same as GetComponentInChildren, but will not recurse into children with component of the StopT type.
- static void [GetNestedComponentsInParents< T >](#) (this Transform t, List< T > list)

Returns all T found between the child transform and its root. Order in List from child to parent, with the root/parent most being last.
- static void [GetNestedComponentsInParents< T, TStop >](#) (this Transform t, List< T > list)

Finds components of type T on supplied transform, and every parent above that node, inclusively stopping on node StopT component.
- static T [GetParentComponent< T >](#) (this Transform t)

Find T on supplied transform or any parent. Unlike GetComponentInParent, GameObjects do not need to be active to be found.

6.153.1 Detailed Description

Tools to replace GetComponent variants that respects nested objects. These are used to find components of a NetworkedObjects without also finding components that belong to parent or child NetworkedObjects.

6.153.2 Member Function Documentation

6.153.2.1 EnsureRootComponentExists< T, TStopOn >()

```
static T EnsureRootComponentExists< T, TStopOn > (
    this Transform transform ) [static]
```

Ensures that a component of type T exists on the root object of the provided transform. If a component of type T does not exist, it is added. The search for the root object stops if a component of type TStopOn is found.

Template Parameters

<i>T</i>	The type of component to ensure exists on the root object.
<i>TStopOn</i>	The type of component that stops the search for the root object.

Parameters

<i>transform</i>	The transform to start the search from.
------------------	---

Returns

The component of type T on the root object. Returns null if no root object is found.

Type Constraints

***T* : Component**

***TStopOn* : Component**

6.153.2.2 FindObjectsOfTypeInOrder< T >() [1/2]

```
static T [] FindObjectsOfTypeInOrder< T > (
    this UnityEngine.SceneManagement.Scene scene,
    bool includeInactive = false ) [static]
```

Find All instances of Component type in a scene. Attempts to respect the hierarchy of the scene objects to produce a more deterministic order. This is a slower operation, and does produce garbage collection.

Type Constraints

***T* : class**

6.153.2.3 FindObjectsOfTypeInOrder< T >() [2/2]

```
static void FindObjectsOfTypeInOrder< T > (  
    this UnityEngine.SceneManagement.Scene scene,  
    List< T > list,  
    bool includeInactive = false ) [static]
```

Find All instances of Component type in a scene. Attempts to respect the hierarchy of the scene objects to produce a more deterministic order. This is a slower operation which should not be run every update.

Template Parameters

<i>T</i>	The type being searched for.
----------	------------------------------

Parameters

<i>scene</i>	Scene to search.
<i>list</i>	Supplied list that will be populated by this find.
<i>includeInactive</i>	Whether results should include inactive components.

Type Constraints

***T* : class**

6.153.2.4 FindObjectsOfTypeInOrder< T, TCast >() [1/2]

```
static TCast [] FindObjectsOfTypeInOrder< T, TCast > (
    this UnityEngine.SceneManagement.Scene scene,
    bool includeInactive = false ) [static]
```

Find All instances of Component type in a scene. Attempts to respect the hierarchy of the scene objects to produce a more deterministic order. This is a slow operation, and does produce garbage collection.

Template Parameters

<i>T</i>	The type being searched for.
<i>TCast</i>	Casts all found objects to this type, and returns collection of this type. Objects that fail cast are excluded.

Parameters

<i>scene</i>	Scene to search.
<i>includeInactive</i>	Whether results should include inactive components.

Type Constraints

***T* : class**

***TCast* : class**

6.153.2.5 FindObjectsOfTypeInOrder< T, TCast >() [2/2]

```
static void FindObjectsOfTypeInOrder< T, TCast > (
    this UnityEngine.SceneManagement.Scene scene,
```

```
List< TCast > list,
bool includeInactive = false ) [static]
```

Find All instances of Component type in a scene. Attempts to respect the hierarchy of the scene objects to produce a more deterministic order. This is a slower operation and should not be run every update.

Template Parameters

<i>T</i>	Type being searched for.
<i>TCast</i>	Type to cast found objects to.

Parameters

<i>scene</i>	Scene to search.
<i>list</i>	Supplied list that will be filled with found objects.
<i>includeInactive</i>	Whether results should include inactive components.

Type Constraints

***T* : class**

***TCast* : class**

6.153.2.6 GetNestedComponentInChildren< T, TStopOn >()

```
static T GetNestedComponentInChildren< T, TStopOn > (
    this Transform t,
    bool includeInactive ) [static]
```

Finds the first component of type T in the children of the provided transform, stopping the search if a component of type TStopOn is found.

Template Parameters

<i>T</i>	The type of component to find.
<i>TStopOn</i>	The type of component that stops the search.

Parameters

<i>t</i>	The transform to start the search from.
<i>includeInactive</i>	Whether to include inactive game objects in the search.

Returns

The first component of type T found. Returns null if no component is found.

Type Constraints

***T* : class**

TStopOn : class

6.153.2.7 GetNestedComponentInParent< T, TStopOn >()

```
static T GetNestedComponentInParent< T, TStopOn > (
    this Transform t ) [static]
```

Same as GetComponentInParent, but will always include inactive objects in search. Will also stop recursing up the hierarchy when the StopOnT is found.

Type Constraints

T : class

TStopOn : class

6.153.2.8 GetNestedComponentInParents< T, TStopOn >()

```
static T GetNestedComponentInParents< T, TStopOn > (
    this Transform t ) [static]
```

Finds the first component of type T in the parents of the provided transform, stopping the search if a component of type TStopOn is found.

Template Parameters

<i>T</i>	The type of component to find.
<i>TStopOn</i>	The type of component that stops the search.

Parameters

<i>t</i>	The transform to start the search from.
----------	---

Returns

The first component of type T found in the parents. Returns null if no component is found or a component of type TStopOn is found.

Type Constraints

T : class

TStopOn : class

6.153.2.9 GetNestedComponentsInChildren< T >()

```
static List<T> GetNestedComponentsInChildren< T > (
    this Transform t,
    List< T > list,
    bool includeInactive = true,
    params System.Type[] stopOn ) [static]
```

Same as GetComponentsInChildren, but will not recurse into children with any component of the types in the stopOn array.

Type Constraints

***T* : class**

6.153.2.10 GetNestedComponentsInChildren< T, TSearch, TStop >()

```
static void GetNestedComponentsInChildren< T, TSearch, TStop > (
    this Transform t,
    bool includeInactive,
    List< T > list ) [static]
```

Same as GetComponentsInChildren, but will not recurse into children with component of the StopT type.

Template Parameters

<i>T</i>	Cast found components to this type. Typically Component, but any other class/interface will work as long as they are assignable from SearchT.
<i>TSearch</i>	Find components of this class or interface type.
<i>TStop</i>	When this component is found, no further recursing will be performed on that node.

Type Constraints

***T* : class**

***TSearch* : class**

6.153.2.11 GetNestedComponentsInChildren< T, TStopOn >()

```
static List<T> GetNestedComponentsInChildren< T, TStopOn > (
    this Transform t,
    List< T > list,
    bool includeInactive = true ) [static]
```

Same as GetComponentsInChildren, but will not recurse into children with component of the StopT type.

Type Constraints

***T* : class**

***TStopOn* : class**

6.153.2.12 GetNestedComponentsInParents< T >()

```
static void GetNestedComponentsInParents< T > (
    this Transform t,
    List< T > list ) [static]
```

Returns all T found between the child transform and its root. Order in List from child to parent, with the root/parent most being last.

Type Constraints

T: ***Component***

6.153.2.13 GetNestedComponentsInParents< T, TStop >()

```
static void GetNestedComponentsInParents< T, TStop > (
    this Transform t,
    List< T > list ) [static]
```

Finds components of type T on supplied transform, and every parent above that node, inclusively stopping on node StopT component.

Type Constraints

T: ***class***

TStop: ***class***

6.153.2.14 GetParentComponent< T >()

```
static T GetParentComponent< T > (
    this Transform t ) [static]
```

Find T on supplied transform or any parent. Unlike GetComponentInParent, GameObjects do not need to be active to be found.

Type Constraints

T: ***Component***

6.154 NetworkArray< T > Struct Template Reference

[Fusion](#) type for networking arrays. Maximum capacity is fixed, and is set with the [CapacityAttribute](#).

Inherits [IEnumerable< T >](#), and [INetworkArray](#).

Classes

- struct [Enumerator](#)
Enumerator for `NetworkArray`.

Public Member Functions

- void [Clear](#) ()
Clears the array, setting all values to default.
- void [CopyFrom](#) (List< T > source, int sourceOffset, int sourceCount)
Copies a range of values from a supplied source list into the `NetworkArray`.
- void [CopyFrom](#) (T[] source, int sourceOffset, int sourceCount)
Copies a range of elements from a source array into the `NetworkArray`.
- void [CopyTo](#) (List< T > list)
Adds each value to the supplied List. This does not clear the list, so values will be appended to the existing list.
- void [CopyTo](#) (`NetworkArray< T >` array)
Copies values to the supplied array.
- void [CopyTo](#) (T[] array, bool throwIfOverflow=true)
Copies values to the supplied array.
- T [Get](#) (int index)
Returns the array value at supplied index.
- [Enumerator GetEnumerator](#) ()
Returns an enumerator that iterates through the collection.
- `IEnumerator< T >` `IEnumerable< T >`. [GetEnumerator](#) ()
- `IEnumerator` `IEnumerable`. [GetEnumerator](#) ()
- `NetworkArray` (byte *array, int length, `IElementReaderWriter< T >` readerWriter)
`NetworkArray` constructor.
- T [Set](#) (int index, T value)
Sets the array value at the supplied index.
- T[] [ToArray](#) ()
Allocates a new array and copies values from this array. For a non-alloc alternative use [CopyTo\(List<T>](#)).
- string [ToListString](#) ()
Returns the elements of this array as a string, with value separated by characters. Specifically for use in the Unity inspector. This is private and only is found by `NetworkBehaviourEditor` using reflection, so do not rename this method.
- `NetworkArrayReadOnly< T >` [ToReadOnly](#) ()
Returns a `NetworkArrayReadOnly` view of this array.
- override string [ToString](#) ()
Returns a string that represents the current object.

Static Public Member Functions

- static implicit `operator NetworkArrayReadOnly< T >` (`NetworkArray< T >` value)
Returns a `NetworkArrayReadOnly` view of this array.

Public Attributes

- byte * **`_array`**
- int **`_length`**
- `IElementReaderWriter< T >` **`_readerWriter`**

Static Public Attributes

- static `StringBuilder` `_stringBuilderCached`

Properties

- int `Length` [get]
The fixed size of the array.
- T `this[int index]` [get, set]
Indexer of array elements.
- object `INetworkArray`. `this[int index]` [get, set]

6.154.1 Detailed Description

`Fusion` type for networking arrays. Maximum capacity is fixed, and is set with the `CapacityAttribute`.

Typical Usage: `[Networked, Capacity(4)]`
`NetworkArray<float> syncedArray => default;`

Optional usage (for `NetworkBehaviours` ONLY - this is not legal in `INetworkStructs`): `[Networked, Capacity(4)]`
`NetworkArray<int> syncedArray { get; } = MakeInitializer(new int[] { 1, 2, 3, 4 });`

Usage for modifying data: `array.Set(123); array[0] = 456;`

Template Parameters

<code>T</code>	<code>T</code> can be a primitive, or an <code>INetworkStruct</code> .
----------------	--

6.154.2 Constructor & Destructor Documentation

6.154.2.1 `NetworkArray()`

```
NetworkArray (
    byte * array,
    int length,
    IElementReaderWriter< T > readerWriter )
```

`NetworkArray` constructor.

6.154.3 Member Function Documentation

6.154.3.1 Clear()

```
void Clear ( )
```

Clears the array, setting all values to default.

6.154.3.2 CopyFrom() [1/2]

```
void CopyFrom (
    List< T > source,
    int sourceOffset,
    int sourceCount )
```

Copies a range of values from a supplied source list into the [NetworkArray](#).

Parameters

<i>source</i>	The source list from which to copy elements.
<i>sourceOffset</i>	The zero-based index in the source list at which copying begins.
<i>sourceCount</i>	The number of elements to copy from the source list.

Exceptions

<i>ArgumentNullException</i>	Thrown when the provided source list is null.
<i>ArgumentException</i>	Thrown when the number of elements to copy is greater than the length of the NetworkArray .
<i>ArgumentOutOfRangeException</i>	Thrown when the sum of <i>sourceOffset</i> and <i>sourceCount</i> is greater than the length of the source list.

6.154.3.3 CopyFrom() [2/2]

```
void CopyFrom (
    T[] source,
    int sourceOffset,
    int sourceCount )
```

Copies a range of elements from a source array into the [NetworkArray](#).

Parameters

<i>source</i>	The source array from which to copy elements. Must not be null.
<i>sourceOffset</i>	The zero-based index in the source array at which copying begins.
<i>sourceCount</i>	The number of elements to copy from the source array.

Exceptions

<i>ArgumentNullException</i>	Thrown when the provided source array is null.
<i>ArgumentException</i>	Thrown when the number of elements to copy is greater than the length of the NetworkArray .
<i>ArgumentOutOfRangeException</i>	Thrown when the sum of sourceOffset and sourceCount is greater than the length of the source array.

6.154.3.4 CopyTo() [1/3]

```
void CopyTo (
    List< T > list )
```

Adds each value to the supplied List. This does not clear the list, so values will be appended to the existing list.

6.154.3.5 CopyTo() [2/3]

```
void CopyTo (
    NetworkArray< T > array )
```

Copies values to the supplied array.

Parameters

<i>array</i>	NetworkArray to copy to.
--------------	--

Exceptions

<i>ArgumentException</i>	Thrown if the supplied array is smaller than this NetworkArray<T> .
--------------------------	---

6.154.3.6 CopyTo() [3/3]

```
void CopyTo (
    T[] array,
    bool throwIfOverflow = true )
```

Copies values to the supplied array.

Parameters

<i>array</i>	Array to copy to.
<i>throwIfOverflow</i>	If true, this method will throw an error if the supplied array is smaller than this NetworkArray<T> . If false, will only copy as many elements as the target array can hold.

6.154.3.7 Get()

```
T Get (
    int index )
```

Returns the array value at supplied index.

6.154.3.8 GetEnumerator()

```
Enumerator GetEnumerator ( )
```

Returns an enumerator that iterates through the collection.

6.154.3.9 operator NetworkArrayReadOnly< T >()

```
static implicit operator NetworkArrayReadOnly< T > (
    NetworkArray< T > value ) [static]
```

Returns a [NetworkArrayReadOnly](#) view of this array.

Parameters

<i>value</i>	NetworkArray to convert.
--------------	--

Returns

[NetworkArrayReadOnly](#) view of this array.

6.154.3.10 Set()

```
T Set (
    int index,
    T value )
```

Sets the array value at the supplied index.

6.154.3.11 ToArray()

```
T [] ToArray ( )
```

Allocates a new array and copies values from this array. For a non-alloc alternative use [CopyTo\(List<T>\)](#).

6.154.3.12 ToListString()

```
string ToListString ( )
```

Returns the elements of this array as a string, with value separated by characters. Specifically for use in the Unity inspector. This is private and only is found by `NetworkBehaviourEditor` using reflection, so do not rename this method.

6.154.3.13 ToReadOnly()

```
NetworkArrayReadOnly<T> ToReadOnly ( )
```

Returns a [NetworkArrayReadOnly](#) view of this array.

6.154.3.14 ToString()

```
override string ToString ( )
```

Returns a string that represents the current object.

6.154.4 Property Documentation

6.154.4.1 Length

```
int Length [get]
```

The fixed size of the array.

6.154.4.2 this[int index]

```
T this[int index] [get], [set]
```

Indexer of array elements.

6.155 `NetworkArray< T >.Enumerator` Struct Reference

[Enumerator](#) for [NetworkArray](#).

Inherits [IEnumerator< T >](#).

Public Member Functions

- void [Dispose](#) ()
Releases all resources used by the [Enumerator](#).
- [Enumerator](#) ([NetworkArray< T > array](#))
Initializes a new instance of the [Enumerator](#) with the specified [NetworkArray](#).
- bool [MoveNext](#) ()
Advances the enumerator to the next element of the collection.
- void [Reset](#) ()
Sets the enumerator to its initial position, which is before the first element in the collection.

Public Attributes

- [NetworkArray< T > _array](#)
- int [_index](#)

Properties

- T [Current](#) [get]
Gets the current element in the collection.
- object [IEnumerator](#). [Current](#) [get]
Gets the current element in the collection.

6.155.1 Detailed Description

[Enumerator](#) for [NetworkArray](#).

6.155.2 Constructor & Destructor Documentation

6.155.2.1 `Enumerator()`

```
Enumerator (  
    NetworkArray< T > array )
```

Initializes a new instance of the [Enumerator](#) with the specified [NetworkArray](#).

Parameters

<code>array</code>	The NetworkArray to enumerate.
--------------------	--

6.155.3 Member Function Documentation

6.155.3.1 Dispose()

```
void Dispose ( )
```

Releases all resources used by the [Enumerator](#).

6.155.3.2 MoveNext()

```
bool MoveNext ( )
```

Advances the enumerator to the next element of the collection.

Returns

true if the enumerator was successfully advanced to the next element; false if the enumerator has passed the end of the collection.

6.155.3.3 Reset()

```
void Reset ( )
```

Sets the enumerator to its initial position, which is before the first element in the collection.

6.155.4 Property Documentation

6.155.4.1 Current [1/2]

```
T Current [get]
```

Gets the current element in the collection.

6.155.4.2 Current [2/2]

```
object IEnumerator. Current [get]
```

Gets the current element in the collection.

6.156 NetworkArrayExtensions Class Reference

Provides extension methods for the [NetworkArray](#) class.

Static Public Member Functions

- static ref T [GetRef< T >](#) (this [NetworkArray< T >](#) array, int index)
Returns a reference to the element at a specified position in the [NetworkArray](#).
- static int [IndexOf< T >](#) (this [NetworkArray< T >](#) array, T elem)
Finds the index of the first occurrence of a specified element in the [NetworkArray](#).

6.156.1 Detailed Description

Provides extension methods for the [NetworkArray](#) class.

6.156.2 Member Function Documentation

6.156.2.1 GetRef< T >()

```
static ref T GetRef< T > (
    this NetworkArray< T > array,
    int index ) [static]
```

Returns a reference to the element at a specified position in the [NetworkArray](#).

Parameters

<i>array</i>	The NetworkArray to search.
<i>index</i>	The zero-based index of the element to get a reference for.

Returns

A reference to the element at the specified position in the [NetworkArray](#).

Type Constraints

T: *unmanaged*

6.156.2.2 IndexOf< T >()

```
static int IndexOf< T > (
    this NetworkArray< T > array,
    T elem ) [static]
```

Finds the index of the first occurrence of a specified element in the [NetworkArray](#).

Parameters

<i>array</i>	The NetworkArray to search.
<i>elem</i>	The element to locate in the NetworkArray .

Returns

The zero-based index of the first occurrence of *elem* within the entire [NetworkArray](#), if found; otherwise, -1.

Type Constraints

T* : *unmanaged

T* : *IEquatable<T>

6.157 NetworkArrayReadOnly< T > Struct Template Reference

Provides a read-only view of a network array.

Public Attributes

- readonly byte * ***_array***
- readonly int ***_length***
- readonly [IElementReaderWriter< T >](#) ***_readerWriter***

Properties

- int [Length](#) [get]
The fixed size of the array.
- T [this\[int index\]](#) [get]
Indexer of array elements.

6.157.1 Detailed Description

Provides a read-only view of a network array.

Template Parameters

<i>T</i>	The type of the elements.
----------	---------------------------

6.157.2 Property Documentation

6.157.2.1 Length

```
int Length [get]
```

The fixed size of the array.

6.157.2.2 this[int index]

```
T this[int index] [get]
```

Indexer of array elements.

6.158 NetworkAssemblyIgnoreAttribute Class Reference

Network Assembly Ignore Attribute

Inherits Attribute.

6.158.1 Detailed Description

Network Assembly Ignore Attribute

6.159 NetworkAssemblyWeavedAttribute Class Reference

Network Assembly Weaved Attribute

Inherits Attribute.

6.159.1 Detailed Description

Network Assembly Weaved Attribute

6.160 NetworkBehaviour Class Reference

Base class for [Fusion](#) network components, which are associated with a [NetworkObject](#).

Inherits [SimulationBehaviour](#), [ISpawned](#), [IDespawned](#), [IElementReaderWriter< NetworkObject >](#), and [IElementReaderWriter< NetworkObject >](#).

Inherited by [HitboxRoot](#), [NetworkMecanimAnimator](#), and [NetworkTRSP](#).

Classes

- struct [ArrayReader](#)
Provides a reader for network arrays of type T.
- struct [BehaviourReader](#)
Provides a reader for network behaviours of type T.
- class [ChangeDetector](#)
Change detector for a [NetworkBehaviour](#)
- struct [DictionaryReader](#)
Provides a reader for network dictionaries with keys of type K and values of type V.
- struct [LinkListReader](#)
Provides a reader for network linked lists of type T.
- struct [PropertyReader](#)
Provides a reader for properties of type T in a network behaviour.

Public Member Functions

- virtual void [CopyBackingFieldsToState](#) (bool firstTime)
Copies the backing fields to the state. This method is meant to be overridden in derived classes.
- void [CopyStateFrom](#) ([NetworkBehaviour](#) source)
Copies entire state of passed in source [NetworkBehaviour](#)
- virtual void [CopyStateToBackingFields](#) ()
Copies the state to the backing fields. This method is meant to be overridden in derived classes.
- virtual void [Despawned](#) ([NetworkRunner](#) runner, bool hasState)
Called before the network object is despawned
- override void [FixedUpdateNetwork](#) ()
Fixed update callback for networked behaviours.
- [ArrayReader](#)< T > [GetArrayReader](#)< T > (string property)
Gets an [ArrayReader](#) for a network array of type T.
- [BehaviourReader](#)< T > [GetBehaviourReader](#)< T > (string property)
Gets a [BehaviourReader](#) for a network behaviour of type T.
- [ChangeDetector](#) [GetChangeDetector](#) ([ChangeDetector.Source](#) source, bool copyInitial=true)
Creates a [ChangeDetector](#) for this network behaviour.
- [DictionaryReader](#)< K, V > [GetDictionaryReader](#)< K, V > (string property)
Gets a [DictionaryReader](#) for a network dictionary with keys of type K and values of type V.
- T? [GetInput](#)< T > ()
- bool [GetInput](#)< T > (out T input)
Returns true if it a valid [INetworkInput](#) can be found for the current simulation tick (Typically this is used in [FixedUpdateNetwork](#)).
- [LinkListReader](#)< T > [GetLinkListReader](#)< T > (string property)
Gets a [LinkListReader](#) for a network linked list of type T.
- int [GetLocalAuthorityMask](#) ()
Gets a bitmask of [AuthorityMasks](#) flags, representing the current local authority over this [NetworkObject](#).
- [PropertyReader](#)< T > [GetPropertyReader](#)< T > (string property)
Gets a [PropertyReader](#) for a property of type T in this network behaviour.
- ref T [ReinterpretState](#)< T > (int offset=0)
Allows read and write access to the internal state buffer
- void [ReplicateTo](#) ([PlayerRef](#) player, bool replicate)
Controls if this network behaviours state is replicated to a player or not
- void [ReplicateToAll](#) (bool replicate)

Sets the default replicated state for this behaviour, this by default is true so calling `ReplicateToAll(true)` does nothing if `ReplicateToAll(false)` hasn't been called before

- void `ResetState ()`
Resets the state of the object to the original state
- virtual void `Spawned ()`
Post spawn callback.
- bool `TryGetSnapshotsBuffers` (out `NetworkBehaviourBuffer` from, out `NetworkBehaviourBuffer` to, out float alpha)
Tries to get the snapshot buffers for this network behaviour.

Static Public Member Functions

- static `ArrayReader< T > GetArrayReader< T >` (Type behaviourType, string property, `IElementReaderWriter< T >` readerWriter=null)
Gets an `ArrayReader` for a network array of type `T` in a network behaviour of a specific type.
- static `BehaviourReader< T > GetBehaviourReader< T >` (`NetworkRunner` runner, Type behaviourType, string property)
Gets a `BehaviourReader` for a network behaviour of type `T`.
- static `BehaviourReader< TProperty > GetBehaviourReader< TBehaviour, TProperty >` (`NetworkRunner` runner, string property)
Gets a `BehaviourReader` for a network behaviour with a specific property of type `TProperty`.
- static `DictionaryReader< K, V > GetDictionaryReader< K, V >` (Type behaviourType, string property, `IElementReaderWriter< K >` keyReaderWriter=null, `IElementReaderWriter< V >` valueReaderWriter=null)
Gets a `DictionaryReader` for a network dictionary with keys of type `K` and values of type `V` in a network behaviour of a specific type.
- static `LinkListReader< T > GetLinkListReader< T >` (Type behaviourType, string property, `IElementReaderWriter< T >` readerWriter=null)
Gets a `LinkListReader` for a network linked list of type `T` in a network behaviour of a specific type.
- static `PropertyReader< T > GetPropertyReader< T >` (Type behaviourType, string property)
Gets a `PropertyReader` for a property of type `T` in a network behaviour of a specific type.
- static `PropertyReader< TProperty > GetPropertyReader< TBehaviour, TProperty >` (string property)
Gets a `PropertyReader` for a property of type `TProperty` in a network behaviour of type `TBehaviour`.
- static `NetworkBehaviourUtils.DictionaryInitializer< K, V > MakeInitializer< K, V >` (`Dictionary< K, V >` dictionary)
This is a special method that is meant to be used only for [Networked] properties inline initialization.
- static `NetworkBehaviourUtils.ArrayInitializer< T > MakeInitializer< T >` (`T[]` array)
This is a special method that is meant to be used only for [Networked] properties inline initialization.
- static `T * MakePtr< T > ()`
Creates a pointer to a value of type `T`. This method is meant to be used only for [Networked] properties inline initialization.
- static `T * MakePtr< T >` (`T defaultValue`)
Creates a pointer to a value of type `T`, initializing it with the provided default value. This method is meant to be used only for [Networked] properties inline initialization.
- static `ref T MakeRef< T > ()`
Creates a reference to a value of type `T`. This method is meant to be used only for [Networked] properties inline initialization.
- static `ref T MakeRef< T >` (`T defaultValue`)
Creates a reference to a value of type `T`, initializing it with the provided default value. This method is meant to be used only for [Networked] properties inline initialization.
- static `int NetworkDeserialize` (`NetworkRunner` runner, byte *data, `ref NetworkBehaviour` result)
Deserializes a `NetworkBehaviour` from a byte array.

- static int [NetworkSerialize](#) ([NetworkRunner](#) runner, [NetworkBehaviour](#) obj, byte *data)
Serializes a [NetworkBehaviour](#) into a byte array.
- static [NetworkBehaviour](#) [NetworkUnwrap](#) ([NetworkRunner](#) runner, [NetworkBehaviourId](#) wrapper)
Converts a [NetworkBehaviourId](#) to a [NetworkBehaviour](#).
- static [NetworkBehaviourId](#) [NetworkWrap](#) ([NetworkBehaviour](#) obj)
Converts a [NetworkBehaviour](#) to a [NetworkBehaviourId](#).
- static [NetworkBehaviourId](#) [NetworkWrap](#) ([NetworkRunner](#) runner, [NetworkBehaviour](#) obj)
Converts a [NetworkBehaviour](#) to a [NetworkBehaviourId](#).
- static implicit operator [NetworkBehaviourId](#) ([NetworkBehaviour](#) behaviour)
Converts [NetworkBehaviour](#) to [NetworkBehaviourId](#)

Public Attributes

- int [offset](#)
Gives access to the offset (in 32 bit words) and count (in 32 bit words) of this behaviour backing data

Protected Member Functions

- virtual bool [ReplicateTo](#) ([PlayerRef](#) player)
Determines whether to replicate the network behaviour to the specified player. This method can be overridden in derived classes to implement custom replication logic.

Properties

- [Tick ChangedTick](#) [get]
The tick the data on this networked behaviour changed
- virtual ? int [DynamicWordCount](#) [get]
Override this value for custom memory allocations. This is for advanced use cases only, and cannot be used if [NetworkedAttribute](#) is used in the derived class.
- bool? [HasInputAuthority](#) [get]
Returns true if the [Simulation.LocalPlayer](#) of the associated [NetworkRunner](#) is the designated as Input Source for this network entity.
- bool? [HasStateAuthority](#) [get]
Returns true if the associated [NetworkRunner](#) is the State Source for this network entity.
- [NetworkBehaviourId? Id](#) [get]
The unique identifier for this network behaviour.
- bool? [IsProxy](#) [get]
Returns true if the associated [NetworkRunner](#) is neither the Input nor State Authority for this network entity. It is recommended to use [!HasStateAuthority](#) or [!HasInputAuthority](#) when possible instead, as this check requires evaluating both authorities - and is therefore less performant than the individual checks.
- [NetworkBehaviourBuffer StateBuffer](#) [get]
Gets the state buffer associated with the network behaviour.
- bool [StateBufferIsValid](#) [get]
Gets a value indicating whether the state buffer is valid.

6.160.1 Detailed Description

Base class for [Fusion](#) network components, which are associated with a [NetworkObject](#).

Derived from [SimulationBehaviour](#), components derived from this class are associated with a [NetworkRunner](#) and [Simulation](#). Components derived from this class are associated with a parent [NetworkObject](#). and can use the [NetworkedAttribute](#) on properties to automate state synchronization, and can use the [RpcAttribute](#) on methods, to automate messaging.

6.160.2 Member Function Documentation

6.160.2.1 CopyBackingFieldsToState()

```
virtual void CopyBackingFieldsToState (
    bool firstTime ) [virtual]
```

Copies the backing fields to the state. This method is meant to be overridden in derived classes.

Parameters

<i>firstTime</i>	Indicates whether this is the first time the method is called.
------------------	--

6.160.2.2 CopyStateFrom()

```
void CopyStateFrom (
    NetworkBehaviour source )
```

Copies entire state of passed in source [NetworkBehaviour](#)

Parameters

<i>source</i>	Source NetworkBehaviour to copy data from
---------------	---

6.160.2.3 CopyStateToBackingFields()

```
virtual void CopyStateToBackingFields ( ) [virtual]
```

Copies the state to the backing fields. This method is meant to be overridden in derived classes.

6.160.2.4 Despawned()

```
virtual void Despawned (
    NetworkRunner runner,
    bool hasState ) [virtual]
```

Called before the network object is despawned

Parameters

<i>runner</i>	The runner that owns the object
<i>hasState</i>	If the state of the behaviour is still accessible

Implements [IDespawned](#).

Reimplemented in [HitboxRoot](#).

6.160.2.5 FixedUpdateNetwork()

```
override void FixedUpdateNetwork ( ) [virtual]
```

Fixed update callback for networked behaviours.

Reimplemented from [SimulationBehaviour](#).

6.160.2.6 GetArrayReader< T >() [1/2]

```
ArrayReader<T> GetArrayReader< T > (
    string property )
```

Gets an [ArrayReader](#) for a network array of type T.

Template Parameters

<i>T</i>	The type of elements in the network array.
----------	--

Parameters

<i>property</i>	The name of the property to be read.
-----------------	--------------------------------------

Returns

An [ArrayReader](#) for the network array of type T.

6.160.2.7 GetArrayReader< T >() [2/2]

```
static ArrayReader<T> GetArrayReader< T > (
    Type behaviourType,
    string property,
    IElementReaderWriter< T > readerWriter = null ) [static]
```

Gets an [ArrayReader](#) for a network array of type T in a network behaviour of a specific type.

Template Parameters

<i>T</i>	The type of elements in the network array.
----------	--

Parameters

<i>behaviourType</i>	The type of the network behaviour.
<i>property</i>	The name of the property to be read.
<i>readerWriter</i>	

Returns

An [ArrayReader](#) for the network array of type T in the network behaviour of the specified type.

6.160.2.8 GetBehaviourReader< T >() [1/2]

```
static BehaviourReader<T> GetBehaviourReader< T > (
    NetworkRunner runner,
    Type behaviourType,
    string property ) [static]
```

Gets a [BehaviourReader](#) for a network behaviour of type T.

Template Parameters

<i>T</i>	The type of the network behaviour.
----------	------------------------------------

Parameters

<i>runner</i>	The NetworkRunner associated with the network behaviour.
<i>behaviourType</i>	The type of the behaviour to be read.
<i>property</i>	The name of the property to be read.

Returns

A [BehaviourReader](#) for the network behaviour of type T.

Type Constraints

***T* : [NetworkBehaviour](#)**

6.160.2.9 GetBehaviourReader< T >() [2/2]

```
BehaviourReader<T> GetBehaviourReader< T > (
    string property )
```

Gets a [BehaviourReader](#) for a network behaviour of type T.

Template Parameters

<i>T</i>	The type of the network behaviour.
----------	------------------------------------

Parameters

<i>property</i>	The name of the property to be read.
-----------------	--------------------------------------

Returns

A [BehaviourReader](#) for the network behaviour of type T.

Type Constraints

***T* : [NetworkBehaviour](#)**

6.160.2.10 GetBehaviourReader< TBehaviour, TProperty >()

```
static BehaviourReader<TProperty> GetBehaviourReader< TBehaviour, TProperty > (
    NetworkRunner runner,
    string property ) [static]
```

Gets a [BehaviourReader](#) for a network behaviour with a specific property of type TProperty.

Template Parameters

<i>TBehaviour</i>	The type of the network behaviour.
<i>TProperty</i>	The type of the property in the network behaviour.

Parameters

<i>runner</i>	The NetworkRunner associated with the network behaviour.
<i>property</i>	The name of the property to be read.

Returns

A [BehaviourReader](#) for the network behaviour with the specific property of type TProperty.

Type Constraints

TBehaviour : *NetworkBehaviour*

TProperty : *NetworkBehaviour*

6.160.2.11 GetChangeDetector()

```
ChangeDetector GetChangeDetector (
    ChangeDetector.Source source,
    bool copyInitial = true )
```

Creates a [ChangeDetector](#) for this network behaviour.

Parameters

<i>source</i>	The source of the change detector.
<i>copyInitial</i>	Indicates whether to copy the initial state of the network behaviour.

Returns

A [ChangeDetector](#) for this network behaviour.

6.160.2.12 GetDictionaryReader< K, V >() [1/2]

```
DictionaryReader<K, V> GetDictionaryReader< K, V > (
    string property )
```

Gets a [DictionaryReader](#) for a network dictionary with keys of type K and values of type V.

Template Parameters

<i>K</i>	The type of keys in the network dictionary.
<i>V</i>	The type of values in the network dictionary.

Parameters

<i>property</i>	The name of the property to be read.
-----------------	--------------------------------------

Returns

A [DictionaryReader](#) for the network dictionary with keys of type K and values of type V.

6.160.2.13 GetDictionaryReader< K, V >() [2/2]

```
static DictionaryReader<K, V> GetDictionaryReader< K, V > (
    Type behaviourType,
    string property,
    IElementReaderWriter< K > keyReaderWriter = null,
    IElementReaderWriter< V > valueReaderWriter = null ) [static]
```

Gets a [DictionaryReader](#) for a network dictionary with keys of type K and values of type V in a network behaviour of a specific type.

Template Parameters

<i>K</i>	The type of keys in the network dictionary.
<i>V</i>	The type of values in the network dictionary.

Parameters

<i>behaviourType</i>	The type of the network behaviour.
<i>property</i>	The name of the property to be read.
<i>keyReaderWriter</i>	
<i>valueReaderWriter</i>	

Returns

A [DictionaryReader](#) for the network dictionary with keys of type K and values of type V in the network behaviour of the specified type.

6.160.2.14 GetInput< T >() [1/2]

```
T? GetInput< T > ( )
```

Template Parameters

<i>T</i>	
----------	--

Type Constraints

T* : *unmanaged

T* : *INetworkInput

6.160.2.15 GetInput< T >() [2/2]

```
bool GetInput< T > (
    out T input )
```

Returns true if it a valid [INetworkInput](#) can be found for the current simulation tick (Typically this is used in [FixedUpdateNetwork](#)).

The returned input struct originates from the [NetworkObject.InputAuthority](#), and if valid contains the inputs supplied by that [PlayerRef](#) for the current simulation tick.

Type Constraints

T* : *unmanaged

T* : *INetworkInput

6.160.2.16 GetLinkListReader< T >() [1/2]

```
LinkListReader<T> GetLinkListReader< T > (
    string property )
```

Gets a [LinkListReader](#) for a network linked list of type T.

Template Parameters

<i>T</i>	The type of elements in the network linked list.
----------	--

Parameters

<i>property</i>	The name of the property to be read.
-----------------	--------------------------------------

Returns

A [LinkListReader](#) for the network linked list of type T.

6.160.2.17 GetLinkListReader< T >() [2/2]

```
static LinkListReader<T> GetLinkListReader< T > (
    Type behaviourType,
    string property,
    IElementReaderWriter< T > readerWriter = null ) [static]
```

Gets a [LinkListReader](#) for a network linked list of type T in a network behaviour of a specific type.

Template Parameters

<i>T</i>	The type of elements in the network linked list.
----------	--

Parameters

<i>behaviourType</i>	The type of the network behaviour.
<i>property</i>	The name of the property to be read.
<i>readerWriter</i>	

Returns

A [LinkedListReader](#) for the network linked list of type *T* in the network behaviour of the specified type.

6.160.2.18 GetLocalAuthorityMask()

```
int GetLocalAuthorityMask ( )
```

Gets a bitmask of [AuthorityMasks](#) flags, representing the current local authority over this [NetworkObject](#).

6.160.2.19 GetPropertyReader< T >() [1/2]

```
PropertyReader<T> GetPropertyReader< T > (
    string property )
```

Gets a [PropertyReader](#) for a property of type *T* in this network behaviour.

Template Parameters

<i>T</i>	The type of the property in the network behaviour. Must be unmanaged.
----------	---

Parameters

<i>property</i>	The name of the property to be read.
-----------------	--------------------------------------

Returns

A [PropertyReader](#) for the property of type *T* in this network behaviour.

Type Constraints

***T* : unmanaged**

6.160.2.20 `GetPropertyReader< T >()` [2/2]

```
static PropertyReader<T> GetPropertyReader< T > (
    Type behaviourType,
    string property ) [static]
```

Gets a [PropertyReader](#) for a property of type T in a network behaviour of a specific type.

Template Parameters

<i>T</i>	The type of the property in the network behaviour. Must be unmanaged.
----------	---

Parameters

<i>behaviourType</i>	The type of the network behaviour.
<i>property</i>	The name of the property to be read.

Returns

A [PropertyReader](#) for the property of type T in the network behaviour of the specified type.

Type Constraints

T* : *unmanaged

6.160.2.21 `GetPropertyReader< TBehaviour, TProperty >()`

```
static PropertyReader<TProperty> GetPropertyReader< TBehaviour, TProperty > (
    string property ) [static]
```

Gets a [PropertyReader](#) for a property of type TProperty in a network behaviour of type TBehaviour.

Template Parameters

<i>TBehaviour</i>	The type of the network behaviour.
<i>TProperty</i>	The type of the property in the network behaviour. Must be unmanaged.

Parameters

<i>property</i>	The name of the property to be read.
-----------------	--------------------------------------

Returns

A [PropertyReader](#) for the property of type TProperty in the network behaviour of type TBehaviour.

Type Constraints

TBehaviour* : *NetworkBehaviour

TProperty : unmanaged

6.160.2.22 MakeInitializer< K, V >()

```
static NetworkBehaviourUtils.DictionaryInitializer<K, V> MakeInitializer< K, V > (
    Dictionary< K, V > dictionary ) [static]
```

This is a special method that is meant to be used only for [Networked] properties inline initialization.

6.160.2.23 MakeInitializer< T >()

```
static NetworkBehaviourUtils.ArrayInitializer<T> MakeInitializer< T > (
    T[] array ) [static]
```

This is a special method that is meant to be used only for [Networked] properties inline initialization.

6.160.2.24 MakePtr< T >() [1/2]

```
static T* MakePtr< T > ( ) [static]
```

Creates a pointer to a value of type T. This method is meant to be used only for [Networked] properties inline initialization.

Template Parameters

<i>T</i>	The type of the value. Must be unmanaged.
----------	---

Returns

A pointer to a value of type T.

Exceptions

<i>NotImplementedException</i>	Thrown because this method is meant to be used only for [Networked] properties inline initialization.
--------------------------------	---

Type Constraints

T : unmanaged

6.160.2.25 MakePtr< T >() [2/2]

```
static T* MakePtr< T > (
    T defaultValue ) [static]
```

Creates a pointer to a value of type T, initializing it with the provided default value. This method is meant to be used only for [Networked] properties inline initialization.

Template Parameters

<i>T</i>	The type of the value. Must be unmanaged.
----------	---

Parameters

<i>defaultValue</i>	The default value to initialize the value with.
---------------------	---

Returns

A pointer to a value of type T.

Exceptions

<i>NotImplementedException</i>	Thrown because this method is meant to be used only for [Networked] properties inline initialization.
--------------------------------	---

Type Constraints

T* : *unmanaged

6.160.2.26 MakeRef< T >() [1/2]

```
static ref T MakeRef< T > ( ) [static]
```

Creates a reference to a value of type T. This method is meant to be used only for [Networked] properties inline initialization.

Template Parameters

<i>T</i>	The type of the value. Must be unmanaged.
----------	---

Returns

A reference to a value of type T.

Exceptions

<i>NotImplementedException</i>	Thrown because this method is meant to be used only for [Networked] properties inline initialization.
--------------------------------	---

Type Constraints

T* : *unmanaged

6.160.2.27 MakeRef< T >() [2/2]

```
static ref T MakeRef< T > (
    T defaultValue ) [static]
```

Creates a reference to a value of type T, initializing it with the provided default value. This method is meant to be used only for [Networked] properties inline initialization.

Template Parameters

<i>T</i>	The type of the value. Must be unmanaged.
----------	---

Parameters

<i>defaultValue</i>	The default value to initialize the value with.
---------------------	---

Returns

A reference to a value of type T.

Exceptions

<i>NotImplementedException</i>	Thrown because this method is meant to be used only for [Networked] properties inline initialization.
--------------------------------	---

Type Constraints

T* : *unmanaged

6.160.2.28 NetworkDeserialize()

```
static int NetworkDeserialize (
    NetworkRunner runner,
    byte * data,
    ref NetworkBehaviour result ) [static]
```

Deserializes a [NetworkBehaviour](#) from a byte array.

Parameters

<i>runner</i>	The NetworkRunner associated with the NetworkBehaviour .
<i>data</i>	The byte pointer to read the serialized data from.
<i>result</i>	The NetworkBehaviour to store the deserialized data in.

Returns

The size of the deserialized data in bytes.

6.160.2.29 NetworkSerialize()

```
static int NetworkSerialize (
    NetworkRunner runner,
    NetworkBehaviour obj,
    byte * data ) [static]
```

Serializes a [NetworkBehaviour](#) into a byte array.

Parameters

<i>runner</i>	
<i>obj</i>	The NetworkBehaviour to be serialized.
<i>data</i>	The byte pointer to write the serialized data to.

Returns

The size of the serialized data in bytes.

6.160.2.30 NetworkUnwrap()

```
static NetworkBehaviour NetworkUnwrap (
    NetworkRunner runner,
    NetworkBehaviourId wrapper ) [static]
```

Converts a [NetworkBehaviourId](#) to a [NetworkBehaviour](#).

Parameters

<i>runner</i>	The NetworkRunner associated with the NetworkBehaviour .
<i>wrapper</i>	The NetworkBehaviourId to be converted.

Returns

The [NetworkBehaviour](#) represented by the [NetworkBehaviourId](#). If the [NetworkBehaviourId](#) is not valid or a [NetworkBehaviour](#) with the [NetworkBehaviourId](#) does not exist, returns null.

6.160.2.31 NetworkWrap() [1/2]

```
static NetworkBehaviourId NetworkWrap (
    NetworkBehaviour obj ) [static]
```

Converts a [NetworkBehaviour](#) to a [NetworkBehaviourId](#).

Parameters

<i>obj</i>	The NetworkBehaviour to be converted.
------------	---

Returns

A [NetworkBehaviourId](#) representing the [NetworkBehaviour](#). If the [NetworkBehaviour](#) is null or its [NetworkObject](#)'s Id is default, returns a default [NetworkBehaviourId](#).

6.160.2.32 NetworkWrap() [2/2]

```
static NetworkBehaviourId NetworkWrap (
    NetworkRunner runner,
    NetworkBehaviour obj ) [static]
```

Converts a [NetworkBehaviour](#) to a [NetworkBehaviourId](#).

Parameters

<i>runner</i>	The NetworkRunner associated with the NetworkBehaviour .
<i>obj</i>	The NetworkBehaviour to be converted.

Returns

A [NetworkBehaviourId](#) representing the [NetworkBehaviour](#). If the [NetworkBehaviour](#) is null or its [NetworkObject](#)'s Id is default, returns a default [NetworkBehaviourId](#).

6.160.2.33 operator NetworkBehaviourId()

```
static implicit operator NetworkBehaviourId (
    NetworkBehaviour behaviour ) [static]
```

Converts [NetworkBehaviour](#) to [NetworkBehaviourId](#)

Parameters

<i>behaviour</i>	NetworkBehaviour to convert
------------------	---

Returns

[NetworkBehaviourId](#) representing the [NetworkBehaviour](#)

6.160.2.34 ReinterpretState< T >()

```
ref T ReinterpretState< T > (  
    int offset = 0 )
```

Allows read and write access to the internal state buffer

Parameters

<i>offset</i>	The offset to generate a ref for, in integer words
---------------	--

Template Parameters

<i>T</i>	The type of the ref to generate
----------	---------------------------------

Returns

Reference to the location in memory defined by offset

Type Constraints

T* : *unmanaged

6.160.2.35 ReplicateTo() [1/2]

```
virtual bool ReplicateTo (  
    PlayerRef player ) [protected], [virtual]
```

Determines whether to replicate the network behaviour to the specified player. This method can be overridden in derived classes to implement custom replication logic.

Parameters

<i>player</i>	The player to potentially replicate the network behaviour to.
---------------	---

Returns

True if the network behaviour should be replicated to the player, false otherwise. The default implementation always returns true.

6.160.2.36 ReplicateTo() [2/2]

```
void ReplicateTo (
    PlayerRef player,
    bool replicate )
```

Controls if this network behaviours state is replicated to a player or not

Parameters

<i>player</i>	The player to change replication status for
<i>replicate</i>	true = replicate, false = don't replicate

6.160.2.37 ReplicateToAll()

```
void ReplicateToAll (
    bool replicate )
```

Sets the default replicated state for this behaviour, this by default is true so calling ReplicateToAll(true) does nothing if ReplicateToAll(false) hasn't been called before

Parameters

<i>replicate</i>	The default state of this behaviour
------------------	-------------------------------------

6.160.2.38 ResetState()

```
void ResetState ( )
```

Resets the state of the object to the original state

6.160.2.39 Spawned()

```
virtual void Spawned ( ) [virtual]
```

Post spawn callback.

Implements [ISpawned](#).

Reimplemented in [NetworkTransform](#), and [NetworkMecanimAnimator](#).

6.160.2.40 TryGetSnapshotsBuffers()

```
bool TryGetSnapshotsBuffers (
    out NetworkBehaviourBuffer from,
    out NetworkBehaviourBuffer to,
    out float alpha )
```

Tries to get the snapshot buffers for this network behaviour.

Parameters

<i>from</i>	The buffer representing the state of the network behaviour at the start of the render frame.
<i>to</i>	The buffer representing the state of the network behaviour at the end of the render frame.
<i>alpha</i>	The interpolation factor between the start and end of the render frame.

Returns

True if the snapshot buffers are valid, false otherwise.

6.160.3 Member Data Documentation

6.160.3.1 offset

```
int offset
```

Gives access to the offset (in 32 bit words) and count (in 32 bit words) of this behaviour backing data

6.160.4 Property Documentation

6.160.4.1 ChangedTick

`Tick ChangedTick [get]`

The tick the data on this networked behaviour changed

6.160.4.2 DynamicWordCount

`virtual ? int DynamicWordCount [get]`

Override this value for custom memory allocations. This is for advanced use cases only, and cannot be used if [NetworkedAttribute](#) is used in the derived class.

6.160.4.3 HasInputAuthority

`bool? HasInputAuthority [get]`

Returns true if the [Simulation.LocalPlayer](#) of the associated [NetworkRunner](#) is the designated as Input Source for this network entity.

6.160.4.4 HasStateAuthority

`bool? HasStateAuthority [get]`

Returns true if the associated [NetworkRunner](#) is the State Source for this network entity.

6.160.4.5 Id

`NetworkBehaviourId? Id [get]`

The unique identifier for this network behaviour.

6.160.4.6 IsProxy

`bool? IsProxy [get]`

Returns true if the associated [NetworkRunner](#) is neither the Input nor State Authority for this network entity. It is recommended to use [!HasStateAuthority](#) or [!HasInputAuthority](#) when possible instead, as this check requires evaluating both authorities - and is therefore less performant than the individual checks.

6.160.4.7 StateBuffer

```
NetworkBehaviourBuffer StateBuffer [get]
```

Gets the state buffer associated with the network behaviour.

6.160.4.8 StateBufferIsValid

```
bool StateBufferIsValid [get]
```

Gets a value indicating whether the state buffer is valid.

6.161 NetworkBehaviour.ArrayReader< T > Struct Template Reference

Provides a reader for network arrays of type T.

Public Member Functions

- [NetworkArrayReadOnly< T > Read \(NetworkBehaviourBuffer first\)](#)
Reads a network array from the provided network behaviour buffer.

6.161.1 Detailed Description

Provides a reader for network arrays of type T.

Template Parameters

<i>T</i>	The type of elements in the network array.
----------	--

6.161.2 Member Function Documentation

6.161.2.1 Read()

```
NetworkArrayReadOnly<T> Read (  
    NetworkBehaviourBuffer first )
```

Reads a network array from the provided network behaviour buffer.

Parameters

<i>first</i>	The network behaviour buffer to read the network array from.
--------------	--

Returns

A read-only view of the network array.

6.162 NetworkBehaviour.BehaviourReader< T > Struct Template Reference

Provides a reader for network behaviours of type T.

Public Member Functions

- [T Read](#) ([NetworkBehaviourBuffer](#) first)
Reads a network behaviour from the provided network behaviour buffer.
- [T Read](#) ([NetworkBehaviourBuffer](#) first, [NetworkBehaviourBuffer](#) second)

Public Attributes

- [T](#)
Reads two network behaviours from the provided network behaviour buffers.

6.162.1 Detailed Description

Provides a reader for network behaviours of type T.

Template Parameters

<i>T</i>	The type of the network behaviour.
----------	------------------------------------

Type Constraints

T: [NetworkBehaviour](#)

6.162.2 Member Function Documentation

6.162.2.1 Read()

`T Read (NetworkBehaviourBuffer first)`

Reads a network behaviour from the provided network behaviour buffer.

Parameters

<i>first</i>	The network behaviour buffer to read the network behaviour from.
--------------	--

Returns

The network behaviour of type T read from the buffer. Returns null if the behaviour is not found.

6.162.3 Member Data Documentation

6.162.3.1 T

T

Reads two network behaviours from the provided network behaviour buffers.

Parameters

<i>first</i>	The first network behaviour buffer to read the network behaviour from.
<i>second</i>	The second network behaviour buffer to read the network behaviour from.

Returns

A tuple containing the two network behaviours of type T read from the buffers.

6.163 NetworkBehaviour.ChangeDetector Class Reference

Change detector for a [NetworkBehaviour](#)

Classes

- struct [Enumerable](#)
Struct representing a collection of changes detected in a [NetworkBehaviour](#).
- struct [Enumerator](#)
[Enumerator](#) for the collection of changes detected in a [NetworkBehaviour](#).

Public Types

- enum class [Source](#)
Enum representing the source of a [NetworkBehaviour](#)'s state.

Public Member Functions

- [Enumerable DetectChanges](#) ([NetworkBehaviour](#) b, bool copyChanges=true)
Detects changes in a [NetworkBehaviour](#) and returns an [Enumerable](#) of the changes.
- [Enumerable DetectChanges](#) ([NetworkBehaviour](#) b, out [NetworkBehaviourBuffer](#) previous, out [NetworkBehaviourBuffer](#) current, bool copyChanges=true)
Detects changes in a [NetworkBehaviour](#) and returns an [Enumerable](#) of the changes.
- void [Init](#) ([NetworkBehaviour](#) networkBehaviour, [Source](#) source, bool copyInitial=true)
Initializes the [ChangeDetector](#) for a given [NetworkBehaviour](#).

6.163.1 Detailed Description

Change detector for a [NetworkBehaviour](#)

This class is used to detect changes in a [NetworkBehaviour](#). It can be used to detect changes in a [NetworkBehaviour](#) between two snapshots, or between the current state and a snapshot.

6.163.2 Member Enumeration Documentation

6.163.2.1 Source

```
enum Source [strong]
```

Enum representing the source of a [NetworkBehaviour](#)'s state.

Enumerator

SimulationState	The state is the current simulation state of the NetworkBehaviour .
SnapshotFrom	The state is from a previous snapshot of the NetworkBehaviour .
SnapshotTo	The state is from a future snapshot of the NetworkBehaviour .

6.163.3 Member Function Documentation

6.163.3.1 DetectChanges() [1/2]

```
Enumerable DetectChanges (
    NetworkBehaviour b,
    bool copyChanges = true )
```

Detects changes in a [NetworkBehaviour](#) and returns an [Enumerable](#) of the changes.

Parameters

<i>b</i>	The NetworkBehaviour instance to detect changes in.
<i>copyChanges</i>	Whether to copy the changes detected. Defaults to true.

Returns

An [Enumerable](#) of the changes detected in the [NetworkBehaviour](#).

6.163.3.2 DetectChanges() [2/2]

```
Enumerable DetectChanges (
    NetworkBehaviour b,
    out NetworkBehaviourBuffer previous,
    out NetworkBehaviourBuffer current,
    bool copyChanges = true )
```

Detects changes in a [NetworkBehaviour](#) and returns an [Enumerable](#) of the changes.

Parameters

<i>b</i>	The NetworkBehaviour instance to detect changes in.
<i>previous</i>	The previous state of the NetworkBehaviour .
<i>current</i>	The current state of the NetworkBehaviour .
<i>copyChanges</i>	Whether to copy the changes detected. Defaults to true.

Returns

An [Enumerable](#) of the changes detected in the [NetworkBehaviour](#).

6.163.3.3 Init()

```
void Init (
    NetworkBehaviour networkBehaviour,
    Source source,
    bool copyInitial = true )
```

Initializes the [ChangeDetector](#) for a given [NetworkBehaviour](#).

Parameters

<i>networkBehaviour</i>	The NetworkBehaviour instance to initialize the ChangeDetector for.
<i>source</i>	The source of the NetworkBehaviour 's state.
<i>copyInitial</i>	Whether to copy the initial state of the NetworkBehaviour . Defaults to true.

6.164 NetworkBehaviour.ChangeDetector.Enumerable Struct Reference

Struct representing a collection of changes detected in a [NetworkBehaviour](#).

Public Member Functions

- [bool Changed](#) (string name)
Checks if a property has changed.
- [Enumerator GetEnumerator](#) ()
Gets an enumerator for the collection of changes.

6.164.1 Detailed Description

Struct representing a collection of changes detected in a [NetworkBehaviour](#).

6.164.2 Member Function Documentation

6.164.2.1 Changed()

```
bool Changed (  
    string name )
```

Checks if a property has changed.

Parameters

<i>name</i>	The name of the property to check.
-------------	------------------------------------

Returns

True if the property has changed, false otherwise.

6.164.2.2 GetEnumerator()

```
Enumerator GetEnumerator ( )
```

Gets an enumerator for the collection of changes.

Returns

An [Enumerator](#) for the collection of changes.

6.165 NetworkBehaviour.ChangeDetector.Enumerator Struct Reference

[Enumerator](#) for the collection of changes detected in a [NetworkBehaviour](#).

Public Member Functions

- bool [MoveNext](#) ()
Advances the enumerator to the next property in the array of changed properties.
- void [Reset](#) ()
Resets the enumerator to its initial state.

Properties

- string [Current](#) [get]
Gets the current property name in the array of changed properties.

6.165.1 Detailed Description

[Enumerator](#) for the collection of changes detected in a [NetworkBehaviour](#).

6.165.2 Member Function Documentation

6.165.2.1 MoveNext()

```
bool MoveNext ( )
```

Advances the enumerator to the next property in the array of changed properties.

Returns

True if the enumerator was successfully advanced to the next property, false if the enumerator has passed the end of the array.

6.165.2.2 Reset()

```
void Reset ( )
```

Resets the enumerator to its initial state.

6.165.3 Property Documentation

6.165.3.1 Current

```
string Current [get]
```

Gets the current property name in the array of changed properties.

6.166 NetworkBehaviour.DictionaryReader< K, V > Struct Template Reference

Provides a reader for network dictionaries with keys of type K and values of type V.

Public Member Functions

- [NetworkDictionaryReadOnly< K, V > Read \(NetworkBehaviourBuffer first\)](#)
Reads a network dictionary from the provided network behaviour buffer.

6.166.1 Detailed Description

Provides a reader for network dictionaries with keys of type K and values of type V.

Template Parameters

<i>K</i>	The type of keys in the network dictionary.
<i>V</i>	The type of values in the network dictionary.

6.166.2 Member Function Documentation

6.166.2.1 Read()

```
NetworkDictionaryReadOnly<K, V> Read (
    NetworkBehaviourBuffer first )
```

Reads a network dictionary from the provided network behaviour buffer.

Parameters

<i>first</i>	The network behaviour buffer to read the network dictionary from.
--------------	---

Returns

A read-only view of the network dictionary.

6.167 NetworkBehaviour.LinkListReader< T > Struct Template Reference

Provides a reader for network linked lists of type T.

Public Member Functions

- [NetworkLinkedListReadOnly< T > Read \(NetworkBehaviourBuffer first\)](#)
Reads a network linked list from the provided network behaviour buffer.

6.167.1 Detailed Description

Provides a reader for network linked lists of type T.

Template Parameters

<i>T</i>	The type of elements in the network linked list.
----------	--

6.167.2 Member Function Documentation

6.167.2.1 Read()

```
NetworkLinkedListReadOnly<T> Read (  
    NetworkBehaviourBuffer first )
```

Reads a network linked list from the provided network behaviour buffer.

Parameters

<i>first</i>	The network behaviour buffer to read the network linked list from.
--------------	--

Returns

A read-only view of the network linked list.

6.168 NetworkBehaviour.PropertyReader< T > Struct Template Reference

Provides a reader for properties of type T in a network behaviour.

Public Member Functions

- [PropertyReader](#) (int *offset*)
Constructs a new PropertyReader with the provided offset.
- [T Read](#) ([NetworkBehaviourBuffer](#) *first*)
Reads a property of type T from the provided network behaviour buffer.
- [T Read](#) ([NetworkBehaviourBuffer](#) *first*, [NetworkBehaviourBuffer](#) *second*)

Public Attributes

- [T](#)
Reads a property of type T from the provided network behaviour buffers.

6.168.1 Detailed Description

Provides a reader for properties of type T in a network behaviour.

Template Parameters

<i>T</i>	The type of the property in the network behaviour. Must be unmanaged.
----------	---

Type Constraints

T: **unmanaged**

6.168.2 Constructor & Destructor Documentation**6.168.2.1 PropertyReader()**

```
PropertyReader (
    int offset )
```

Constructs a new [PropertyReader](#) with the provided offset.

Parameters

<i>offset</i>	The offset of the property in the network behaviour buffer.
---------------	---

6.168.3 Member Function Documentation

6.168.3.1 Read()

```
T Read (
    NetworkBehaviourBuffer first )
```

Reads a property of type T from the provided network behaviour buffer.

Parameters

<i>first</i>	The network behaviour buffer to read the property from.
--------------	---

Returns

The property of type T read from the buffer.

6.168.4 Member Data Documentation

6.168.4.1 T

T

Reads a property of type T from the provided network behaviour buffers.

Parameters

<i>first</i>	The first network behaviour buffer to read the property from.
<i>second</i>	The second network behaviour buffer to read the property from.

Returns

A tuple containing the property of type T read from the first and second buffers.

6.169 NetworkBehaviourBuffer Struct Reference

Provides low level access to data buffers that can be read using a NetworkBehaviour.Reader

Public Member Functions

- float [Read](#) ([NetworkBehaviour.PropertyTypeReader](#)< float > reader)
Reads a float property from the buffer using the provided PropertyReader.
- Quaternion [Read](#) ([NetworkBehaviour.PropertyTypeReader](#)< Quaternion > reader)
Reads a Quaternion property from the buffer using the provided PropertyReader.
- Vector2 [Read](#) ([NetworkBehaviour.PropertyTypeReader](#)< Vector2 > reader)
Reads a Vector2 property from the buffer using the provided PropertyReader.
- Vector3 [Read](#) ([NetworkBehaviour.PropertyTypeReader](#)< Vector3 > reader)
Reads a Vector3 property from the buffer using the provided PropertyReader.
- Vector4 [Read](#) ([NetworkBehaviour.PropertyTypeReader](#)< Vector4 > reader)
Reads a Vector4 property from the buffer using the provided PropertyReader.
- T [Read](#)< T > ([NetworkBehaviour.BehaviourReader](#)< T > reader)
Reads a [NetworkBehaviour](#) from the buffer using the provided BehaviourReader.
- T [Read](#)< T > ([NetworkBehaviour.PropertyTypeReader](#)< T > reader)
Reads a property from the buffer using the provided PropertyReader.
- unsafe T [ReinterpretState](#)< T > (int offset=0)
Reinterprets the state of the buffer at a given offset as a specific type.

Static Public Member Functions

- static implicit [operator bool](#) ([NetworkBehaviourBuffer](#) buffer)
Implicit conversion operator to bool. This allows a [NetworkBehaviourBuffer](#) instance to be used in conditions directly.

Properties

- int [Length](#) [get]
Gets the length of the buffer.
- int [this\[int index\]](#) [get]
Indexer to get the value at a specific index in the buffer.
- [Tick Tick](#) [get]
Gets the tick at which the buffer was created.
- bool [Valid](#) [get]
Gets a value indicating whether the buffer is valid. A buffer is considered valid if the pointer to the start of the buffer is not null and the length of the buffer is greater than 0.

6.169.1 Detailed Description

Provides low level access to data buffers that can be read using a [NetworkBehaviour.Reader](#)

6.169.2 Member Function Documentation

6.169.2.1 operator bool()

```
static implicit operator bool (
    NetworkBehaviourBuffer buffer ) [static]
```

Implicit conversion operator to bool. This allows a [NetworkBehaviourBuffer](#) instance to be used in conditions directly.

Parameters

<i>buffer</i>	The NetworkBehaviourBuffer instance to convert.
---------------	---

Returns

True if the buffer is valid, false otherwise.

6.169.2.2 Read() [1/5]

```
float Read (
    NetworkBehaviour.PropertyTypeReader< float > reader )
```

Reads a float property from the buffer using the provided PropertyReader.

Parameters

<i>reader</i>	The PropertyReader to use for reading the float property.
---------------	---

Returns

The read float property.

6.169.2.3 Read() [2/5]

```
Quaternion Read (
    NetworkBehaviour.PropertyTypeReader< Quaternion > reader )
```

Reads a Quaternion property from the buffer using the provided PropertyReader.

Parameters

<i>reader</i>	The PropertyReader to use for reading the Quaternion property.
---------------	--

Returns

The read Quaternion property.

6.169.2.4 Read() [3/5]

```
Vector2 Read (
    NetworkBehaviour.PropertyTypeReader< Vector2 > reader )
```

Reads a Vector2 property from the buffer using the provided PropertyReader.

Parameters

<i>reader</i>	The PropertyReader to use for reading the Vector2 property.
---------------	---

Returns

The read Vector2 property.

6.169.2.5 Read() [4/5]

```
Vector3 Read (  
    NetworkBehaviour.PropertyReader< Vector3 > reader )
```

Reads a Vector3 property from the buffer using the provided PropertyReader.

Parameters

<i>reader</i>	The PropertyReader to use for reading the Vector3 property.
---------------	---

Returns

The read Vector3 property.

6.169.2.6 Read() [5/5]

```
Vector4 Read (  
    NetworkBehaviour.PropertyReader< Vector4 > reader )
```

Reads a Vector4 property from the buffer using the provided PropertyReader.

Parameters

<i>reader</i>	The PropertyReader to use for reading the Vector4 property.
---------------	---

Returns

The read Vector4 property.

6.169.2.7 Read< T >() [1/2]

```
T Read< T > (  
    NetworkBehaviour.BehaviourReader< T > reader )
```

Reads a [NetworkBehaviour](#) from the buffer using the provided BehaviourReader.

Template Parameters

<i>T</i>	The type of NetworkBehaviour to read. Must be a subclass of NetworkBehaviour .
----------	--

Parameters

<i>reader</i>	The BehaviourReader to use for reading the NetworkBehaviour .
---------------	---

Returns

The read [NetworkBehaviour](#).

Type Constraints

***T* : [NetworkBehaviour](#)**

6.169.2.8 Read< T >() [2/2]

```
T Read< T > (
    NetworkBehaviour.PropertyReader< T > reader )
```

Reads a property from the buffer using the provided PropertyReader.

Template Parameters

<i>T</i>	The type of the property to read. Must be unmanaged.
----------	--

Parameters

<i>reader</i>	The PropertyReader to use for reading the property.
---------------	---

Returns

The read property.

Type Constraints

***T* : [unmanaged](#)**

6.169.2.9 ReinterpretState< T >()

```
unsafe T ReinterpretState< T > (
    int offset = 0 )
```

Reinterprets the state of the buffer at a given offset as a specific type.

Template Parameters

<i>T</i>	The type to reinterpret the state as. Must be unmanaged.
----------	--

Parameters

<i>offset</i>	The offset at which to start reinterpreting. Defaults to 0.
---------------	---

Returns

The state of the buffer at the given offset, reinterpreted as the specified type.

Type Constraints

T: *unmanaged*

6.169.3 Property Documentation

6.169.3.1 Length

```
int Length [get]
```

Gets the length of the buffer.

6.169.3.2 this[int index]

```
int this[int index] [get]
```

Indexer to get the value at a specific index in the buffer.

Parameters

<i>index</i>	The index to get the value from.
--------------	----------------------------------

Returns

The value at the specified index in the buffer.

6.169.3.3 Tick

```
Tick Tick [get]
```


Gets the tick at which the buffer was created.

6.169.3.4 Valid

```
bool Valid [get]
```

Gets a value indicating whether the buffer is valid. A buffer is considered valid if the pointer to the start of the buffer is not null and the length of the buffer is greater than 0.

6.170 NetworkBehaviourBufferInterpolator Struct Reference

The [NetworkBehaviourBufferInterpolator](#) struct is used to interpolate between two [NetworkBehaviourBuffer](#) instances. This is a read-only, ref struct, meaning it cannot be boxed and it can only be used on the stack.

Public Member Functions

- float [Angle](#) ([NetworkBehaviour.PropertyReader](#)< [Angle](#) > property)
Gets the interpolated angle of a property.
- float [Angle](#) (string property)
Gets the interpolated angle of a property.
- bool [Bool](#) ([NetworkBehaviour.PropertyReader](#)< bool > property)
Gets the interpolated boolean value of a property.
- bool [Bool](#) (string property)
Gets the interpolated boolean value of a property.
- float [Float](#) ([NetworkBehaviour.PropertyReader](#)< float > property)
Gets the interpolated float value of a property.
- float [Float](#) (string property)
Gets the interpolated float value of a property.
- int [Int](#) ([NetworkBehaviour.PropertyReader](#)< int > property)
Gets the interpolated integer value of a property.
- int [Int](#) (string property)
Gets the interpolated integer value of a property.
- [NetworkBehaviourBufferInterpolator](#) ([NetworkBehaviour](#) nb)
Constructor for the [NetworkBehaviourBufferInterpolator](#) struct.
- Quaternion [Quaternion](#) ([NetworkBehaviour.PropertyReader](#)< Quaternion > property)
Gets the interpolated Quaternion value of a property.
- Quaternion [Quaternion](#) (string property)
Gets the interpolated Quaternion value of a property.
- T [Select](#)< T > ([NetworkBehaviour.PropertyReader](#)< T > property)
Selects the interpolated value of a property.
- T [Select](#)< T > (string property)
Selects the interpolated value of a property by its name.
- Vector2 [Vector2](#) ([NetworkBehaviour.PropertyReader](#)< Vector2 > property)
Gets the interpolated Vector2 value of a property.
- Vector2 [Vector2](#) (string property)
Gets the interpolated Vector2 value of a property.

- Vector3 [Vector3](#) ([NetworkBehaviour.PropertyReader](#)< Vector3 > property)
Gets the interpolated Vector3 value of a property.
- Vector3 [Vector3](#) (string property)
Gets the interpolated Vector3 value of a property.
- Vector4 [Vector4](#) ([NetworkBehaviour.PropertyReader](#)< Vector4 > property)
Gets the interpolated Vector4 value of a property.
- Vector4 [Vector4](#) (string property)
Gets the interpolated Vector4 value of a property.

Static Public Member Functions

- static implicit [operator bool](#) ([NetworkBehaviourBufferInterpolator](#) i)
Implicit conversion operator to bool. This allows a [NetworkBehaviourBufferInterpolator](#) instance to be used in conditions directly.

Public Attributes

- readonly float [Alpha](#)
The interpolation factor, ranging from 0 to 1. This value is used to interpolate between the "from" and "to" states.
- readonly [NetworkBehaviour](#) [Behaviour](#)
The [NetworkBehaviour](#) instance that this interpolator is associated with.
- readonly [NetworkBehaviourBuffer](#) [From](#)
The [NetworkBehaviourBuffer](#) instance representing the "from" state for interpolation.
- readonly [NetworkBehaviourBuffer](#) [To](#)
The [NetworkBehaviourBuffer](#) instance representing the "to" state for interpolation.
- readonly bool [Valid](#)
A value indicating whether this interpolator is valid. An interpolator is considered valid if it has successfully retrieved the "from" and "to" buffers.

6.170.1 Detailed Description

The [NetworkBehaviourBufferInterpolator](#) struct is used to interpolate between two [NetworkBehaviourBuffer](#) instances. This is a read-only, ref struct, meaning it cannot be boxed and it can only be used on the stack.

6.170.2 Constructor & Destructor Documentation

6.170.2.1 [NetworkBehaviourBufferInterpolator](#)()

```
NetworkBehaviourBufferInterpolator (  
    NetworkBehaviour nb )
```

Constructor for the [NetworkBehaviourBufferInterpolator](#) struct.

Parameters

<i>nb</i>	The NetworkBehaviour instance that this interpolator is associated with.
-----------	--

6.170.3 Member Function Documentation

6.170.3.1 Angle() [1/2]

```
float Angle (
    NetworkBehaviour.PropertyReader< Angle > property )
```

Gets the interpolated angle of a property.

Parameters

<i>property</i>	The PropertyReader for the property to get the angle of.
-----------------	--

Returns

The interpolated angle of the property.

6.170.3.2 Angle() [2/2]

```
float Angle (
    string property )
```

Gets the interpolated angle of a property.

Parameters

<i>property</i>	The name of the property to get the angle of.
-----------------	---

Returns

The interpolated angle of the property.

6.170.3.3 Bool() [1/2]

```
bool Bool (
    NetworkBehaviour.PropertyReader< bool > property )
```

Gets the interpolated boolean value of a property.

Parameters

<i>property</i>	The PropertyReader for the property to get the boolean value of.
-----------------	--

Returns

The interpolated boolean value of the property.

6.170.3.4 Bool() [2/2]

```
bool Bool (
    string property )
```

Gets the interpolated boolean value of a property.

Parameters

<i>property</i>	The name of the property to get the boolean value of.
-----------------	---

Returns

The interpolated boolean value of the property.

6.170.3.5 Float() [1/2]

```
float Float (
    NetworkBehaviour.PropertyReader< float > property )
```

Gets the interpolated float value of a property.

Parameters

<i>property</i>	The PropertyReader for the property to get the float value of.
-----------------	--

Returns

The interpolated float value of the property.

6.170.3.6 Float() [2/2]

```
float Float (
    string property )
```

Gets the interpolated float value of a property.

Parameters

<i>property</i>	The name of the property to get the float value of.
-----------------	---

Returns

The interpolated float value of the property.

6.170.3.7 Int() [1/2]

```
int Int (
    NetworkBehaviour.PropertyReader< int > property )
```

Gets the interpolated integer value of a property.

Parameters

<i>property</i>	The PropertyReader for the property to get the integer value of.
-----------------	--

Returns

The interpolated integer value of the property.

6.170.3.8 Int() [2/2]

```
int Int (
    string property )
```

Gets the interpolated integer value of a property.

Parameters

<i>property</i>	The name of the property to get the integer value of.
-----------------	---

Returns

The interpolated integer value of the property.

6.170.3.9 operator bool()

```
static implicit operator bool (
    NetworkBehaviourBufferInterpolator i ) [static]
```

Implicit conversion operator to bool. This allows a [NetworkBehaviourBufferInterpolator](#) instance to be used in conditions directly.

Parameters

<i>i</i>	The NetworkBehaviourBufferInterpolator instance to convert.
----------	---

Returns

True if the interpolator is valid, false otherwise.

6.170.3.10 Quaternion() [1/2]

```
Quaternion Quaternion (
    NetworkBehaviour.PropertyReader< Quaternion > property )
```

Gets the interpolated Quaternion value of a property.

Parameters

<i>property</i>	The PropertyReader for the property to get the Quaternion value of.
-----------------	---

Returns

The interpolated Quaternion value of the property.

6.170.3.11 Quaternion() [2/2]

```
Quaternion Quaternion (
    string property )
```

Gets the interpolated Quaternion value of a property.

Parameters

<i>property</i>	The name of the property to get the Quaternion value of.
-----------------	--

Returns

The interpolated Quaternion value of the property.

6.170.3.12 Select< T >() [1/2]

```
T Select< T > (
    NetworkBehaviour.PropertyReader< T > property )
```

Selects the interpolated value of a property.

Template Parameters

<i>T</i>	The type of the property to select. Must be unmanaged.
----------	--

Parameters

<i>property</i>	The PropertyReader for the property to select.
-----------------	--

Returns

The interpolated value of the property.

Type Constraints

T* : *unmanaged

6.170.3.13 Select< T >() [2/2]

```
T Select< T > (
    string property )
```

Selects the interpolated value of a property by its name.

Template Parameters

<i>T</i>	The type of the property to select. Must be unmanaged.
----------	--

Parameters

<i>property</i>	The name of the property to select.
-----------------	-------------------------------------

Returns

The interpolated value of the property.

Type Constraints

T* : *unmanaged

6.170.3.14 Vector2() [1/2]

```
Vector2 Vector2 (
    NetworkBehaviour.PropertyReader< Vector2 > property )
```

Gets the interpolated Vector2 value of a property.

Parameters

<i>property</i>	The PropertyReader for the property to get the Vector2 value of.
-----------------	--

Returns

The interpolated Vector2 value of the property.

6.170.3.15 Vector2() [2/2]

```
Vector2 Vector2 (
    string property )
```

Gets the interpolated Vector2 value of a property.

Parameters

<i>property</i>	The name of the property to get the Vector2 value of.
-----------------	---

Returns

The interpolated Vector2 value of the property.

6.170.3.16 Vector3() [1/2]

```
Vector3 Vector3 (
    NetworkBehaviour.PropertyReader< Vector3 > property )
```

Gets the interpolated Vector3 value of a property.

Parameters

<i>property</i>	The PropertyReader for the property to get the Vector3 value of.
-----------------	--

Returns

The interpolated Vector3 value of the property.

6.170.3.17 Vector3() [2/2]

```
Vector3 Vector3 (
    string property )
```

Gets the interpolated Vector3 value of a property.

Parameters

<i>property</i>	The name of the property to get the Vector3 value of.
-----------------	---

Returns

The interpolated Vector3 value of the property.

6.170.3.18 Vector4() [1/2]

```
Vector4 Vector4 (
    NetworkBehaviour.PropertyReader< Vector4 > property )
```

Gets the interpolated Vector4 value of a property.

Parameters

<i>property</i>	The PropertyReader for the property to get the Vector4 value of.
-----------------	--

Returns

The interpolated Vector4 value of the property.

6.170.3.19 Vector4() [2/2]

```
Vector4 Vector4 (
    string property )
```

Gets the interpolated Vector4 value of a property.

Parameters

<i>property</i>	The name of the property to get the Vector4 value of.
-----------------	---

Returns

The interpolated Vector4 value of the property.

6.170.4 Member Data Documentation

6.170.4.1 Alpha

readonly float Alpha

The interpolation factor, ranging from 0 to 1. This value is used to interpolate between the "from" and "to" states.

6.170.4.2 Behaviour

readonly [NetworkBehaviour](#) Behaviour

The [NetworkBehaviour](#) instance that this interpolator is associated with.

6.170.4.3 From

readonly [NetworkBehaviourBuffer](#) From

The [NetworkBehaviourBuffer](#) instance representing the "from" state for interpolation.

6.170.4.4 To

readonly [NetworkBehaviourBuffer](#) To

The [NetworkBehaviourBuffer](#) instance representing the "to" state for interpolation.

6.170.4.5 Valid

readonly bool Valid

A value indicating whether this interpolator is valid. An interpolator is considered valid if it has successfully retrieved the "from" and "to" buffers.

6.171 NetworkBehaviourId Struct Reference

Represents the unique identifier for a [NetworkBehaviour](#) instance.

Inherits [INetworkStruct](#), and [IEquatable< NetworkBehaviourId >](#).

Public Member Functions

- bool [Equals](#) ([NetworkBehaviourId](#) other)

Checks if this [NetworkBehaviourId](#) is equal to another [NetworkBehaviourId](#). Two [NetworkBehaviourIds](#) are equal if their [Objects](#) and [Behaviours](#) are equal.
- override bool [Equals](#) (object obj)

Checks if this [NetworkBehaviourId](#) is equal to another object. The object is considered equal if it is a [NetworkBehaviourId](#) and its [Object](#) and [Behaviour](#) are equal to this [NetworkBehaviourId](#)'s.
- override int [GetHashCode](#) ()

Returns a hash code for this [NetworkBehaviourId](#). The hash code is computed based on the [Object](#)'s hash code and the [Behaviour](#).
- override string [ToString](#) ()

Returns a string representation of the [NetworkBehaviourId](#). The string representation is in the format: [[Object](#)←→ :{[Object](#)}, [Behaviour](#):{[Behaviour](#)}].

Static Public Member Functions

- static bool [operator!=](#) ([NetworkBehaviourId](#) a, [NetworkBehaviourId](#) b)

Determines whether two [NetworkBehaviourId](#) instances are not equal. Two [NetworkBehaviourId](#) instances are considered not equal if their [Objects](#) or [Behaviours](#) are not equal.
- static bool [operator==](#) ([NetworkBehaviourId](#) a, [NetworkBehaviourId](#) b)

Determines whether two [NetworkBehaviourId](#) instances are equal. Two [NetworkBehaviourId](#) instances are considered equal if their [Objects](#) and [Behaviours](#) are equal.

Public Attributes

- int [Behaviour](#)

The identifier for the behaviour within the object.
- [NetworkId](#) [Object](#)

The [NetworkId](#) of the object this behaviour belongs to.

Static Public Attributes

- const int [SIZE](#) = [NetworkId](#).[SIZE](#) + sizeof(int)

Size of the [NetworkBehaviourId](#) structure in bytes.

Properties

- bool [IsValid](#) [get]

Checks if the [NetworkBehaviourId](#) is valid. A [NetworkBehaviourId](#) is valid if its [Object](#) is valid and its [Behaviour](#) is non-negative.
- static [NetworkBehaviourId](#) [None](#) [get]

Returns a new [NetworkBehaviourId](#) with default values.

6.171.1 Detailed Description

Represents the unique identifier for a [NetworkBehaviour](#) instance.

6.171.2 Member Function Documentation

6.171.2.1 Equals() [1/2]

```
bool Equals (
    NetworkBehaviourId other )
```

Checks if this [NetworkBehaviourId](#) is equal to another [NetworkBehaviourId](#). Two [NetworkBehaviourId](#)s are equal if their Objects and Behaviours are equal.

Parameters

<i>other</i>	The other NetworkBehaviourId to compare with.
--------------	---

Returns

True if the [NetworkBehaviourId](#)s are equal, false otherwise.

6.171.2.2 Equals() [2/2]

```
override bool Equals (
    object obj )
```

Checks if this [NetworkBehaviourId](#) is equal to another object. The object is considered equal if it is a [NetworkBehaviourId](#) and its Object and [Behaviour](#) are equal to this [NetworkBehaviourId](#)'s.

Parameters

<i>obj</i>	The object to compare with.
------------	-----------------------------

Returns

True if the object is a [NetworkBehaviourId](#) and is equal to this, false otherwise.

6.171.2.3 GetHashCode()

```
override int GetHashCode ( )
```

Returns a hash code for this [NetworkBehaviourId](#). The hash code is computed based on the Object's hash code and the [Behaviour](#).

Returns

A hash code for this [NetworkBehaviourId](#).

6.171.2.4 operator"!=()

```
static bool operator!= (
    NetworkBehaviourId a,
    NetworkBehaviourId b ) [static]
```

Determines whether two [NetworkBehaviourId](#) instances are not equal. Two [NetworkBehaviourId](#) instances are considered not equal if their Objects or Behaviours are not equal.

Parameters

<i>a</i>	The first NetworkBehaviourId to compare.
<i>b</i>	The second NetworkBehaviourId to compare.

Returns

True if the [NetworkBehaviourId](#) instances are not equal, false otherwise.

6.171.2.5 operator=="()

```
static bool operator== (
    NetworkBehaviourId a,
    NetworkBehaviourId b ) [static]
```

Determines whether two [NetworkBehaviourId](#) instances are equal. Two [NetworkBehaviourId](#) instances are considered equal if their Objects and Behaviours are equal.

Parameters

<i>a</i>	The first NetworkBehaviourId to compare.
<i>b</i>	The second NetworkBehaviourId to compare.

Returns

True if the [NetworkBehaviourId](#) instances are equal, false otherwise.

6.171.2.6 ToString()

```
override string ToString ( )
```

Returns a string representation of the [NetworkBehaviourId](#). The string representation is in the format: [Object←: {Object}, Behaviour: {Behaviour}].

Returns

A string representation of the [NetworkBehaviourId](#).

6.171.3 Member Data Documentation

6.171.3.1 Behaviour

```
int Behaviour
```

The identifier for the behaviour within the object.

6.171.3.2 Object

```
NetworkId Object
```

The [NetworkId](#) of the object this behaviour belongs to.

6.171.3.3 SIZE

```
const int SIZE = NetworkId.SIZE + sizeof(int) [static]
```

Size of the [NetworkBehaviourId](#) structure in bytes.

6.171.4 Property Documentation

6.171.4.1 IsValid

```
bool IsValid [get]
```

Checks if the [NetworkBehaviourId](#) is valid. A [NetworkBehaviourId](#) is valid if its [Object](#) is valid and its [Behaviour](#) is non-negative.

6.171.4.2 None

`NetworkBehaviourId` None [static], [get]

Returns a new `NetworkBehaviourId` with default values.

6.172 NetworkBehaviourUtils Class Reference

This static class provides utility methods for working with `NetworkBehaviour` objects.

Classes

- struct `ArrayInitializer`
A utility structure for initializing `NetworkArray` and `NetworkLinkedList` with inline initialization.
- struct `DictionaryInitializer`
A utility structure for initializing `NetworkDictionary` with inline initialization.
- struct `MetaData`
This structure holds metadata for a `NetworkBehaviour` object.

Static Public Member Functions

- static `T[] CloneArray< T > (T[] array)`
Creates a new array that is a clone of the specified array.
- static void `CopyFromNetworkArray< T > (NetworkArray< T > networkArray, ref T[] dstArray)`
Copies the values from a `NetworkArray` to a destination array.
- static void `CopyFromNetworkDictionary< D, K, V > (NetworkDictionary< K, V > networkDictionary, ref D dictionary)`
Copies the values from a `NetworkDictionary` to a destination dictionary.
- static void `CopyFromNetworkList< T > (NetworkLinkedList< T > networkList, ref T[] dstArray)`
Copies the values from a `NetworkLinkedList` to a destination array.
- static `MetaData GetMetaData (Type type)`
Retrieves the metadata for a given type.
- static int `GetRpcStaticIndexOrThrow (string key)`
Retrieves the index of a static RPC (Remote Procedure Call) based on its key.
- static int `GetStaticWordCount (Type type)`
Retrieves the static word count for a given type. If the word count for the type has not been computed yet, it is computed and stored.
- static int `GetWordCount (NetworkBehaviour behaviour)`
Retrieves the word count for a given `NetworkBehaviour`. If the `NetworkBehaviour` has a dynamic word count, it is returned. Otherwise, the static word count for the type of the `NetworkBehaviour` is returned.
- static bool `HasStaticWordCount (Type type)`
Checks if a given type has a static word count. A type has a static word count if it is a subclass of `NetworkBehaviour` and its word count is non-negative.
- static void `InitializeNetworkArray< T > (NetworkArray< T > networkArray, T[] sourceArray, string name)`
Initializes a `NetworkArray` with the values from a source array.
- static void `InitializeNetworkDictionary< D, K, V > (NetworkDictionary< K, V > networkDictionary, D dictionary, string name)`
Initializes a `NetworkDictionary` with the values from a source dictionary.

- static void [InitializeNetworkList](#)< T > ([NetworkLinkedList](#)< T > networkList, T[] sourceArray, string name)
Initializes a [NetworkLinkedList](#) with the values from a source array.
- static void [InternalOnDestroy](#) ([SimulationBehaviour](#) obj)
This method is not meant to be called directly. Calls are injected by the Weaver.
- static void [InternalOnDisable](#) ([SimulationBehaviour](#) obj)
This method is not meant to be called directly. Calls are injected by the Weaver.
- static void [InternalOnEnable](#) ([SimulationBehaviour](#) obj)
This method is not meant to be called directly. Calls are injected by the Weaver.
- static [SerializableDictionary](#)< K, V > [MakeSerializableDictionary](#)< K, V > ([Dictionary](#)< K, V > dictionary)
Wraps a [Dictionary](#) in a [SerializableDictionary](#).
- static void [NotifyLocalSimulationNotAllowedToSendRpc](#) (string rpc, [NetworkObject](#) obj, int sources)
Logs an error message indicating that a local simulation is not allowed to send a specific RPC.
- static void [NotifyLocalTargetedRpcCulled](#) ([PlayerRef](#) player, string methodName)
Logs a warning message indicating that a local targeted RPC was culled.
- static void [NotifyNetworkUnwrapFailed](#)< T > (T wrapper, Type valueType)
Logs a warning message indicating that a network unwrap operation failed.
- static void [NotifyNetworkWrapFailed](#)< T > (T value)
Logs a warning message indicating that a network wrap operation failed.
- static void [NotifyNetworkWrapFailed](#)< T > (T value, Type wrapperType)
Logs a warning message indicating that a network wrap operation failed for a specific wrapper type.
- static void [NotifyRpcPayloadSizeExceeded](#) (string rpc, int size)
Logs an error message indicating that the target of a Remote Procedure Call (RPC) payload is too large.
- static void [NotifyRpcTargetUnreachable](#) ([PlayerRef](#) player, string rpc)
Logs an error message indicating that the target of a Remote Procedure Call (RPC) is not reachable.
- static void [RegisterMetaData](#) (Type type)
Registers the metadata for a given type. This method checks if the type has an override for the "ReplicateTo" method and stores this information in the metadata. If the metadata for the type already exists, this method does nothing.
- static void [RegisterRpcInvokeDelegates](#) (Type type)
Registers the RPC (Remote Procedure Call) invoke delegates for a given type. This method is only available when the FUSION_UNITY compilation symbol is defined.
- static bool [ShouldRegisterRpcInvokeDelegates](#) (Type type)
Determines whether the RPC (Remote Procedure Call) invoke delegates should be registered for a given type.
- static void [ThrowIfBehaviourNotInitialized](#) ([NetworkBehaviour](#) behaviour)
Checks if the [NetworkBehaviour](#) object is initialized and throws an exception if it is not.
- static bool [TryGetRpcInvokeDelegateArray](#) (Type type, out [RpcInvokeData](#)[] delegates)
Tries to get the RPC (Remote Procedure Call) invoke delegate array for a given type.
- static bool [TryGetRpcStaticInvokeDelegate](#) (int index, out [RpcStaticInvokeDelegate](#) del)
Tries to get the static RPC (Remote Procedure Call) invoke delegate based on its index.

Static Public Attributes

- static bool [InvokeRpc](#) = false
A static field that determines whether to invoke RPC (Remote Procedure Call). When set to true, RPCs are invoked. When set to false, RPCs are not invoked. Default value is false.

6.172.1 Detailed Description

This static class provides utility methods for working with [NetworkBehaviour](#) objects.

6.172.2 Member Function Documentation

6.172.2.1 CloneArray< T >()

```
static T [] CloneArray< T > (
    T[] array ) [static]
```

Creates a new array that is a clone of the specified array.

Template Parameters

<i>T</i>	The type of the elements in the array.
----------	--

Parameters

<i>array</i>	The array to clone.
--------------	---------------------

Returns

A new array that is a clone of the specified array. If the specified array is null, an empty array is returned.

6.172.2.2 CopyFromNetworkArray< T >()

```
static void CopyFromNetworkArray< T > (
    NetworkArray< T > networkArray,
    ref T[] dstArray ) [static]
```

Copies the values from a [NetworkArray](#) to a destination array.

Template Parameters

<i>T</i>	The type of the elements in the NetworkArray and destination array.
----------	---

Parameters

<i>networkArray</i>	The NetworkArray from which to copy the values.
<i>dstArray</i>	The destination array to which to copy the values.

If the length of the destination array is not equal to the length of the [NetworkArray](#), a new array of the correct length is created and assigned to the destination array.

Type Constraints

T* : *unmanaged

6.172.2.3 CopyFromNetworkDictionary< D, K, V >()

```
static void CopyFromNetworkDictionary< D, K, V > (
    NetworkDictionary< K, V > networkDictionary,
    ref D dictionary ) [static]
```

Copies the values from a [NetworkDictionary](#) to a destination dictionary.

Template Parameters

<i>D</i>	The type of the destination dictionary. Must implement IDictionary{K, V} and have a parameterless constructor.
<i>K</i>	The type of the keys in the NetworkDictionary and destination dictionary. Must be unmanaged.
<i>V</i>	The type of the values in the NetworkDictionary and destination dictionary. Must be unmanaged.

Parameters

<i>networkDictionary</i>	The NetworkDictionary from which to copy the values.
<i>dictionary</i>	The destination dictionary to which to copy the values. If the destination dictionary is null, a new dictionary of type D is created.

Type Constraints

***D* : IDictionary**

***D* : K**

***D* : V**

***D* : new()**

***K* : unmanaged**

***V* : unmanaged**

6.172.2.4 CopyFromNetworkList< T >()

```
static void CopyFromNetworkList< T > (
    NetworkLinkedList< T > networkList,
    ref T[] dstArray ) [static]
```

Copies the values from a [NetworkLinkedList](#) to a destination array.

Template Parameters

<i>T</i>	The type of the elements in the NetworkLinkedList and destination array.
----------	--

Parameters

<i>networkList</i>	The NetworkLinkedList from which to copy the values.
<i>dstArray</i>	The destination array to which to copy the values. If the length of the destination array is not equal to the count of the NetworkLinkedList , a new array of the correct length is created and assigned to the destination array.

Type Constraints

T : unmanaged

6.172.2.5 GetMetaData()

```
static MetaData GetMetaData (
    Type type ) [static]
```

Retrieves the metadata for a given type.

Parameters

<i>type</i>	The type for which to retrieve the metadata.
-------------	--

Returns

The metadata for the given type if it exists; otherwise, the default value.

6.172.2.6 GetRpcStaticIndexOrThrow()

```
static int GetRpcStaticIndexOrThrow (
    string key ) [static]
```

Retrieves the index of a static RPC (Remote Procedure Call) based on its key.

Parameters

<i>key</i>	The key of the static RPC.
------------	----------------------------

Returns

The index of the static RPC if it exists.

Exceptions

<i>System.ArgumentOutOfRangeException</i>	Thrown when the static RPC does not exist.
---	--

6.172.2.7 GetStaticWordCount()

```
static int GetStaticWordCount (
    Type type ) [static]
```

Retrieves the static word count for a given type. If the word count for the type has not been computed yet, it is computed and stored.

Parameters

<i>type</i>	The type for which to retrieve the static word count.
-------------	---

Returns

The static word count for the given type.

Exceptions

<i>System.ArgumentException</i>	Thrown when the provided type is not a subclass of NetworkBehaviour .
<i>System.InvalidOperationException</i>	Thrown when the provided type does not have a weaved attribute or its word count is negative.

6.172.2.8 GetWordCount()

```
static int GetWordCount (
    NetworkBehaviour behaviour ) [static]
```

Retrieves the word count for a given [NetworkBehaviour](#). If the [NetworkBehaviour](#) has a dynamic word count, it is returned. Otherwise, the static word count for the type of the [NetworkBehaviour](#) is returned.

Parameters

<i>behaviour</i>	The NetworkBehaviour for which to retrieve the word count.
------------------	--

Returns

The word count for the given [NetworkBehaviour](#).

Exceptions

<i>System.InvalidOperationException</i>	Thrown when the dynamic word count or the static word count is negative.
---	--

6.172.2.9 HasStaticWordCount()

```
static bool HasStaticWordCount (
    Type type ) [static]
```

Checks if a given type has a static word count. A type has a static word count if it is a subclass of [NetworkBehaviour](#) and its word count is non-negative.

Parameters

<i>type</i>	The type to check.
-------------	--------------------

Returns

True if the type has a static word count, false otherwise.

Exceptions

<i>System.ArgumentException</i>	Thrown when the provided type is not a subclass of NetworkBehaviour .
<i>System.InvalidOperationException</i>	Thrown when the provided type does not have a weaved attribute.

6.172.2.10 InitializeNetworkArray< T >()

```
static void InitializeNetworkArray< T > (
    NetworkArray< T > networkArray,
    T[] sourceArray,
    string name ) [static]
```

Initializes a [NetworkArray](#) with the values from a source array.

Template Parameters

<i>T</i>	The type of the elements in the NetworkArray and source array.
----------	--

Parameters

<i>networkArray</i>	The NetworkArray to initialize.
<i>sourceArray</i>	The source array from which to copy the values.
<i>name</i>	The name of the NetworkArray for logging purposes.

If the length of the source array is greater than the length of the [NetworkArray](#), a warning is logged and only the first elements up to the length of the [NetworkArray](#) are copied.

Type Constraints

T* : *unmanaged

6.172.2.11 InitializeNetworkDictionary< D, K, V >()

```
static void InitializeNetworkDictionary< D, K, V > (
    NetworkDictionary< K, V > networkDictionary,
    D dictionary,
    string name ) [static]
```

Initializes a [NetworkDictionary](#) with the values from a source dictionary.

Template Parameters

<i>D</i>	The type of the source dictionary. Must implement IDictionary{K, V}.
<i>K</i>	The type of the keys in the NetworkDictionary and source dictionary. Must be unmanaged.
<i>V</i>	The type of the values in the NetworkDictionary and source dictionary. Must be unmanaged.

Parameters

<i>networkDictionary</i>	The NetworkDictionary to initialize.
<i>dictionary</i>	The source dictionary from which to copy the values.
<i>name</i>	The name of the NetworkDictionary for logging purposes.

If the count of the source dictionary is greater than the capacity of the [NetworkDictionary](#), a warning is logged and only the first elements up to the capacity of the [NetworkDictionary](#) are copied.

Type Constraints

***D* : IDictionary**

***D* : K**

***D* : V**

***K* : unmanaged**

***V* : unmanaged**

6.172.2.12 InitializeNetworkList< T >()

```
static void InitializeNetworkList< T > (
    NetworkLinkedList< T > networkList,
    T[] sourceArray,
    string name ) [static]
```

Initializes a [NetworkLinkedList](#) with the values from a source array.

Template Parameters

<i>T</i>	The type of the elements in the NetworkLinkedList and source array. Must be unmanaged.
----------	--

Parameters

<i>networkList</i>	The NetworkLinkedList to initialize.
<i>sourceArray</i>	The source array from which to copy the values.
<i>name</i>	The name of the NetworkLinkedList for logging purposes.

If the length of the source array is greater than the capacity of the [NetworkLinkedList](#), a warning is logged and only the first elements up to the capacity of the [NetworkLinkedList](#) are copied.

Type Constraints

T* : *unmanaged

6.172.2.13 InternalOnDestroy()

```
static void InternalOnDestroy (
    SimulationBehaviour obj ) [static]
```

This method is not meant to be called directly. Calls are injected by the Weaver.

Parameters

<i>obj</i>	SimulationBehaviour object.
------------	---

6.172.2.14 InternalOnDisable()

```
static void InternalOnDisable (
    SimulationBehaviour obj ) [static]
```

This method is not meant to be called directly. Calls are injected by the Weaver.

Parameters

<i>obj</i>	SimulationBehaviour object.
------------	---

6.172.2.15 InternalOnEnable()

```
static void InternalOnEnable (
    SimulationBehaviour obj ) [static]
```

This method is not meant to be called directly. Calls are injected by the Weaver.

Parameters

<i>obj</i>	SimulationBehaviour object.
------------	-----------------------------

6.172.2.16 MakeSerializableDictionary< K, V >()

```
static SerializableDictionary<K, V> MakeSerializableDictionary< K, V > (
    Dictionary< K, V > dictionary ) [static]
```

Wraps a Dictionary in a [SerializableDictionary](#).

Template Parameters

<i>K</i>	The type of the keys in the dictionary. Must be unmanaged.
<i>V</i>	The type of the values in the dictionary. Must be unmanaged.

Parameters

<i>dictionary</i>	The dictionary to wrap.
-------------------	-------------------------

Returns

A [SerializableDictionary](#) that wraps the specified dictionary.

Type Constraints

***K* : unmanaged**

***V* : unmanaged**

6.172.2.17 NotifyLocalSimulationNotAllowedToSendRpc()

```
static void NotifyLocalSimulationNotAllowedToSendRpc (
    string rpc,
    NetworkObject obj,
    int sources ) [static]
```

Logs an error message indicating that a local simulation is not allowed to send a specific RPC.

Parameters

<i>rpc</i>	The name of the RPC that was attempted.
<i>obj</i>	The network object on which the RPC was attempted.
<i>sources</i>	The sources from which the RPC was attempted.

6.172.2.18 NotifyLocalTargetedRpcCulled()

```
static void NotifyLocalTargetedRpcCulled (
    PlayerRef player,
    string methodName ) [static]
```

Logs a warning message indicating that a local targeted RPC was culled.

Parameters

<i>player</i>	The player reference for which the RPC was culled.
<i>methodName</i>	The name of the method that was culled.

6.172.2.19 NotifyNetworkUnwrapFailed< T >()

```
static void NotifyNetworkUnwrapFailed< T > (
    T wrapper,
    Type valueType ) [static]
```

Logs a warning message indicating that a network unwrap operation failed.

Template Parameters

<i>T</i>	The type of the wrapper that failed to be unwrapped.
----------	--

Parameters

<i>wrapper</i>	The wrapper that failed to be unwrapped.
<i>valueType</i>	The type that was attempted to be obtained from the unwrap operation.

6.172.2.20 NotifyNetworkWrapFailed< T >() [1/2]

```
static void NotifyNetworkWrapFailed< T > (
    T value ) [static]
```

Logs a warning message indicating that a network wrap operation failed.

Template Parameters

<i>T</i>	The type of the value that failed to be wrapped.
----------	--

Parameters

<i>value</i>	The value that failed to be wrapped.
--------------	--------------------------------------

6.172.2.21 NotifyNetworkWrapFailed< T >() [2/2]

```
static void NotifyNetworkWrapFailed< T > (
    T value,
    Type wrapperType ) [static]
```

Logs a warning message indicating that a network wrap operation failed for a specific wrapper type.

Template Parameters

<i>T</i>	The type of the value that failed to be wrapped.
----------	--

Parameters

<i>value</i>	The value that failed to be wrapped.
<i>wrapperType</i>	The type that was attempted to be used as a wrapper.

6.172.2.22 NotifyRpcPayloadSizeExceeded()

```
static void NotifyRpcPayloadSizeExceeded (
    string rpc,
    int size ) [static]
```

Logs an error message indicating that the target of a Remote Procedure Call (RPC) payload is too large.

Parameters

<i>size</i>	Size of the payload.
<i>rpc</i>	The name of the RPC that was attempted.

6.172.2.23 NotifyRpcTargetUnreachable()

```
static void NotifyRpcTargetUnreachable (
    PlayerRef player,
    string rpc ) [static]
```

Logs an error message indicating that the target of a Remote Procedure Call (RPC) is not reachable.

Parameters

<i>player</i>	The player reference that is not reachable.
<i>rpc</i>	The name of the RPC that was attempted.

6.172.2.24 RegisterMetaData()

```
static void RegisterMetaData (
    Type type ) [static]
```

Registers the metadata for a given type. This method checks if the type has an override for the "ReplicateTo" method and stores this information in the metadata. If the metadata for the type already exists, this method does nothing.

Parameters

<i>type</i>	The type for which to register the metadata.
-------------	--

6.172.2.25 RegisterRpcInvokeDelegates()

```
static void RegisterRpcInvokeDelegates (
    Type type ) [static]
```

Registers the RPC (Remote Procedure Call) invoke delegates for a given type. This method is only available when the FUSION_UNITY compilation symbol is defined.

Parameters

<i>type</i>	The type for which to register the RPC invoke delegates.
-------------	--

Exceptions

<i>System.ArgumentException</i>	Thrown when the provided type is not a subclass of NetworkBehaviour .
<i>System.InvalidOperationException</i>	Thrown when the provided type does not have a weaved attribute or its word count is negative.

6.172.2.26 ShouldRegisterRpcInvokeDelegates()

```
static bool ShouldRegisterRpcInvokeDelegates (
    Type type ) [static]
```

Determines whether the RPC (Remote Procedure Call) invoke delegates should be registered for a given type.

Parameters

<i>type</i>	The type for which to check the registration of RPC invoke delegates.
-------------	---

Returns

True if the RPC invoke delegates should be registered for the type, false otherwise.

6.172.2.27 ThrowIfBehaviourNotInitialized()

```
static void ThrowIfBehaviourNotInitialized (
    NetworkBehaviour behaviour ) [static]
```

Checks if the [NetworkBehaviour](#) object is initialized and throws an exception if it is not.

Parameters

<i>behaviour</i>	The NetworkBehaviour object to check.
------------------	---

Exceptions

<i>System.InvalidOperationException</i>	Thrown when the NetworkBehaviour object is not initialized.
---	---

6.172.2.28 TryGetRpcInvokeDelegateArray()

```
static bool TryGetRpcInvokeDelegateArray (
    Type type,
    out RpcInvokeData[] delegates ) [static]
```

Tries to get the RPC (Remote Procedure Call) invoke delegate array for a given type.

Parameters

<i>type</i>	The type for which to get the RPC invoke delegate array.
<i>delegates</i>	When this method returns, contains the RPC invoke delegate array if the type has one; otherwise, null.

Returns

true if the type has an RPC invoke delegate array; otherwise, false.

6.172.2.29 TryGetRpcStaticInvokeDelegate()

```
static bool TryGetRpcStaticInvokeDelegate (
    int index,
    out RpcStaticInvokeDelegate del ) [static]
```

Tries to get the static RPC (Remote Procedure Call) invoke delegate based on its index.

Parameters

<i>index</i>	The index of the static RPC.
<i>del</i>	When this method returns, contains the static RPC invoke delegate if it exists; otherwise, default.

Returns

True if the static RPC invoke delegate exists; otherwise, false.

6.172.3 Member Data Documentation

6.172.3.1 InvokeRpc

```
bool InvokeRpc = false [static]
```

A static field that determines whether to invoke RPC (Remote Procedure Call). When set to true, RPCs are invoked. When set to false, RPCs are not invoked. Default value is false.

6.173 NetworkBehaviourUtils.ArrayInitializer< T > Struct Template Reference

A utility structure for initializing [NetworkArray](#) and [NetworkLinkedList](#) with inline initialization.

Static Public Member Functions

- static implicit [operator NetworkArray< T > \(ArrayInitializer< T > arr\)](#)
Implicitly converts an [ArrayInitializer](#) to a [NetworkArray](#).
- static implicit [operator NetworkLinkedList< T > \(ArrayInitializer< T > arr\)](#)
Implicitly converts an [ArrayInitializer](#) to a [NetworkLinkedList](#).

6.173.1 Detailed Description

A utility structure for initializing [NetworkArray](#) and [NetworkLinkedList](#) with inline initialization.

Template Parameters

<i>T</i>	The type of the elements in the NetworkArray and NetworkLinkedList .
----------	--

6.173.2 Member Function Documentation

6.173.2.1 operator NetworkArray< T >()

```
static implicit operator NetworkArray< T > (  
    ArrayInitializer< T > arr ) [static]
```

Implicitly converts an [ArrayInitializer](#) to a [NetworkArray](#).

Parameters

<i>arr</i>	The ArrayInitializer to convert.
------------	--

Returns

A [NetworkArray](#) initialized with the values from the [ArrayInitializer](#).

Exceptions

<i>System.NotImplementedException</i>	Thrown always as this method is meant to be used only for [Networked] properties inline initialization.
---------------------------------------	---

6.173.2.2 operator NetworkLinkedList< T >()

```
static implicit operator NetworkLinkedList< T > (  
    ArrayInitializer< T > arr ) [static]
```

Implicitly converts an [ArrayInitializer](#) to a [NetworkLinkedList](#).

Parameters

<i>arr</i>	The ArrayInitializer to convert.
------------	--

Returns

A [NetworkLinkedList](#) initialized with the values from the [ArrayInitializer](#).

Exceptions

<i>System.NotImplementedException</i>	Thrown always as this method is meant to be used only for [Networked] properties inline initialization.
---------------------------------------	---

6.174 NetworkBehaviourUtils.DictionaryInitializer< K, V > Struct Template Reference

A utility structure for initializing [NetworkDictionary](#) with inline initialization.

Static Public Member Functions

- static implicit [operator NetworkDictionary< K, V > \(DictionaryInitializer< K, V > arr\)](#)
Implicitly converts a [DictionaryInitializer](#) to a [NetworkDictionary](#).

6.174.1 Detailed Description

A utility structure for initializing [NetworkDictionary](#) with inline initialization.

Template Parameters

<i>K</i>	The type of the keys in the NetworkDictionary .
<i>V</i>	The type of the values in the NetworkDictionary .

6.174.2 Member Function Documentation

6.174.2.1 operator NetworkDictionary< K, V >()

```
static implicit operator NetworkDictionary< K, V > (  
    DictionaryInitializer< K, V > arr ) [static]
```

Implicitly converts a [DictionaryInitializer](#) to a [NetworkDictionary](#).

Parameters

<i>arr</i>	The DictionaryInitializer to convert.
------------	---

Returns

A [NetworkDictionary](#) initialized with the values from the [DictionaryInitializer](#).

Exceptions

<code>System.NotImplementedException</code>	Thrown always as this method is meant to be used only for [Networked] properties inline initialization.
---	---

6.175 NetworkBehaviourUtils.MetaData Struct Reference

This structure holds metadata for a [NetworkBehaviour](#) object.

6.175.1 Detailed Description

This structure holds metadata for a [NetworkBehaviour](#) object.

6.176 NetworkBehaviourWeavedAttribute Class Reference

Network [Behaviour](#) Weaved Attribute

Inherits [Attribute](#).

Public Member Functions

- [NetworkBehaviourWeavedAttribute](#) (int wordCount)
NetworkBehaviourWeavedAttribute Constructor

Properties

- int [WordCount](#) [get]
NetworkBehaviour Word Count

6.176.1 Detailed Description

Network [Behaviour](#) Weaved Attribute

6.176.2 Constructor & Destructor Documentation

6.176.2.1 NetworkBehaviourWeavedAttribute()

```
NetworkBehaviourWeavedAttribute (  
    int wordCount )
```

[NetworkBehaviourWeavedAttribute](#) Constructor

Parameters

<code>wordCount</code>	WordCount
------------------------	---------------------------

6.176.3 Property Documentation

6.176.3.1 WordCount

```
int WordCount [get]
```

[NetworkBehaviour](#) Word Count

6.177 NetworkBool Struct Reference

Represents a boolean value that can be networked.

Inherits [INetworkStruct](#), and [IEquatable< NetworkBool >](#).

Public Member Functions

- bool [Equals](#) ([NetworkBool](#) other)
Determines whether the specified [NetworkBool](#) is equal to the current [NetworkBool](#).
- override bool [Equals](#) (object obj)
Determines whether the specified object is equal to the current [NetworkBool](#).
- override int [GetHashCode](#) ()
Serves as the default hash function.
- [NetworkBool](#) (bool value)
Initializes a new instance of the [NetworkBool](#) struct with the specified value.
- override string [ToString](#) ()
Returns a string that represents the current [NetworkBool](#).

Static Public Member Functions

- static implicit [operator bool](#) ([NetworkBool](#) val)
Defines an implicit conversion of a [NetworkBool](#) to a bool.
- static implicit [operator NetworkBool](#) (bool val)
Defines an implicit conversion of a bool to a [NetworkBool](#).

Public Attributes

- `int _value`

Static Public Attributes

- const int [SIZE](#) = 4

The size of the [NetworkBool](#) structure in bytes.

6.177.1 Detailed Description

Represents a boolean value that can be networked.

6.177.2 Constructor & Destructor Documentation

6.177.2.1 NetworkBool()

```
NetworkBool (
    bool value )
```

Initializes a new instance of the [NetworkBool](#) struct with the specified value.

Parameters

<i>value</i>	The boolean value.
--------------	--------------------

6.177.3 Member Function Documentation

6.177.3.1 Equals() [1/2]

```
bool Equals (
    NetworkBool other )
```

Determines whether the specified [NetworkBool](#) is equal to the current [NetworkBool](#).

Parameters

<i>other</i>	The NetworkBool to compare with the current NetworkBool .
--------------	---

Returns

true if the specified [NetworkBool](#) is equal to the current [NetworkBool](#); otherwise, false.

6.177.3.2 Equals() [2/2]

```
override bool Equals (  
    object obj )
```

Determines whether the specified object is equal to the current [NetworkBool](#).

Parameters

<i>obj</i>	The object to compare with the current NetworkBool .
------------	--

Returns

true if the specified object is equal to the current [NetworkBool](#); otherwise, false.

6.177.3.3 GetHashCode()

```
override int GetHashCode ( )
```

Serves as the default hash function.

Returns

A hash code for the current [NetworkBool](#).

6.177.3.4 operator bool()

```
static implicit operator bool (  
    NetworkBool val ) [static]
```

Defines an implicit conversion of a [NetworkBool](#) to a bool.

Parameters

<i>val</i>	The NetworkBool to convert.
------------	---

6.177.3.5 operator NetworkBool()

```
static implicit operator NetworkBool (  
    bool val ) [static]
```

Defines an implicit conversion of a bool to a [NetworkBool](#).

Parameters

<code>val</code>	The bool to convert.
------------------	----------------------

6.177.3.6 ToString()

```
override string ToString ( )
```

Returns a string that represents the current [NetworkBool](#).

Returns

A string that represents the current [NetworkBool](#).

6.177.4 Member Data Documentation

6.177.4.1 SIZE

```
const int SIZE = 4 [static]
```

The size of the [NetworkBool](#) structure in bytes.

6.178 NetworkButtons Struct Reference

Represents a set of buttons that can be networked.

Inherits [INetworkStruct](#), and [IEquatable< NetworkButtons >](#).

Public Member Functions

- bool [Equals](#) ([NetworkButtons](#) other)
Determines whether the specified [NetworkButtons](#) is equal to the current [NetworkButtons](#).
- override bool [Equals](#) (object obj)
Determines whether the specified object is equal to the current [NetworkButtons](#).
- override int [GetHashCode](#) ()
Serves as the default hash function.
- [NetworkButtons](#) [GetPressed](#) ([NetworkButtons](#) previous)
Gets the buttons that were pressed since the previous state.
- [NetworkButtons](#) [GetPressedOrReleased](#) ([NetworkButtons](#) previous)
- [NetworkButtons](#) [GetReleased](#) ([NetworkButtons](#) previous)
Gets the buttons that were released since the previous state.
- bool [IsSet](#) (int button)

- Checks if the button at the specified index is set.*

 - bool `IsSet< T >` (T button)

Checks if the button of the specified enum type is set.
- `NetworkButtons` (int buttons)
 - Initializes a new instance of the `NetworkButtons` struct with the specified buttons state.*
- void `Set` (int button, bool state)
 - Sets the button at the specified index to the specified state.*
- void `Set< T >` (T button, bool state)
 - Sets the button of the specified enum type to the specified state.*
- void `SetAllDown` ()
 - Sets all buttons to down.*
- void `SetAllUp` ()
 - Sets all buttons to up.*
- void `SetDown` (int button)
 - Sets the button at the specified index to down.*
- void `SetDown< T >` (T button)
 - Sets the button of the specified enum type to down.*
- void `SetUp` (int button)
 - Sets the button at the specified index to up.*
- void `SetUp< T >` (T button)
 - Sets the button of the specified enum type to up.*
- bool `WasPressed` (`NetworkButtons` previous, int button)
 - Checks if the button at the specified index was pressed since the previous state.*
- bool `WasPressed< T >` (`NetworkButtons` previous, T button)
 - Checks if the button of the specified enum type was pressed since the previous state.*
- bool `WasReleased` (`NetworkButtons` previous, int button)
 - Checks if the button at the specified index was released since the previous state.*
- bool `WasReleased< T >` (`NetworkButtons` previous, T button)
 - Checks if the button of the specified enum type was released since the previous state.*

Public Attributes

- int `_bits`
- `NetworkButtons`
 - Gets the buttons that were pressed or released since the previous state.*

Properties

- int `Bits` [get]
 - Gets the bits representing the state of the buttons.*

6.178.1 Detailed Description

Represents a set of buttons that can be networked.

6.178.2 Constructor & Destructor Documentation

6.178.2.1 NetworkButtons()

```
NetworkButtons (  
    int buttons )
```

Initializes a new instance of the [NetworkButtons](#) struct with the specified buttons state.

Parameters

<i>buttons</i>	The integer representing the state of the buttons.
----------------	--

6.178.3 Member Function Documentation

6.178.3.1 Equals() [1/2]

```
bool Equals (  
    NetworkButtons other )
```

Determines whether the specified [NetworkButtons](#) is equal to the current [NetworkButtons](#).

Parameters

<i>other</i>	The NetworkButtons to compare with the current NetworkButtons .
--------------	---

Returns

true if the specified [NetworkButtons](#) is equal to the current [NetworkButtons](#); otherwise, false.

6.178.3.2 Equals() [2/2]

```
override bool Equals (  
    object obj )
```

Determines whether the specified object is equal to the current [NetworkButtons](#).

Parameters

<i>obj</i>	The object to compare with the current NetworkButtons .
------------	---

Returns

true if the specified object is equal to the current [NetworkButtons](#); otherwise, false.

6.178.3.3 GetHashCode()

```
override int GetHashCode ( )
```

Serves as the default hash function.

Returns

A hash code for the current [NetworkButtons](#).

6.178.3.4 GetPressed()

```
NetworkButtons GetPressed (   
    NetworkButtons previous )
```

Gets the buttons that were pressed since the previous state.

Parameters

<i>previous</i>	The previous state of the buttons.
-----------------	------------------------------------

Returns

The buttons that were pressed.

6.178.3.5 GetReleased()

```
NetworkButtons GetReleased (   
    NetworkButtons previous )
```

Gets the buttons that were released since the previous state.

Parameters

<i>previous</i>	The previous state of the buttons.
-----------------	------------------------------------

Returns

The buttons that were released.

6.178.3.6 IsSet()

```
bool IsSet (
    int button )
```

Checks if the button at the specified index is set.

Parameters

<i>button</i>	The index of the button to check.
---------------	-----------------------------------

Returns

true if the button is set; otherwise, false.

6.178.3.7 IsSet< T >()

```
bool IsSet< T > (
    T button )
```

Checks if the button of the specified enum type is set.

Template Parameters

<i>T</i>	The enum type of the button.
----------	------------------------------

Parameters

<i>button</i>	The button of type T to check.
---------------	--------------------------------

Returns

true if the button is set; otherwise, false.

Type Constraints

T* : *unmanaged

T* : *Enum

6.178.3.8 Set()

```
void Set (
    int button,
    bool state )
```

Sets the button at the specified index to the specified state.

Parameters

<i>button</i>	The index of the button to set.
<i>state</i>	The state to set the button to.

6.178.3.9 Set< T >()

```
void Set< T > (
    T button,
    bool state )
```

Sets the button of the specified enum type to the specified state.

Template Parameters

<i>T</i>	The enum type of the button.
----------	------------------------------

Parameters

<i>button</i>	The button of type T to set.
<i>state</i>	The state to set the button to.

Type Constraints

T* : *unmanaged

T* : *Enum

6.178.3.10 SetAllDown()

```
void SetAllDown ( )
```

Sets all buttons to down.

6.178.3.11 SetAllUp()

```
void SetAllUp ( )
```

Sets all buttons to up.

6.178.3.12 SetDown()

```
void SetDown (
    int button )
```

Sets the button at the specified index to down.

Parameters

<i>button</i>	The index of the button to set to down.
---------------	---

6.178.3.13 SetDown< T >()

```
void SetDown< T > (  
    T button )
```

Sets the button of the specified enum type to down.

Template Parameters

<i>T</i>	The enum type of the button.
----------	------------------------------

Parameters

<i>button</i>	The button of type T to set to down.
---------------	--------------------------------------

Type Constraints

T* : *unmanaged

T* : *Enum

6.178.3.14 SetUp()

```
void SetUp (  
    int button )
```

Sets the button at the specified index to up.

Parameters

<i>button</i>	The index of the button to set to up.
---------------	---------------------------------------

6.178.3.15 SetUp< T >()

```
void SetUp< T > (  
    T button )
```

Sets the button of the specified enum type to up.

Template Parameters

<i>T</i>	The enum type of the button.
----------	------------------------------

Parameters

<i>button</i>	The button of type T to set to up.
---------------	------------------------------------

Type Constraints

T* : *unmanaged

T* : *Enum

6.178.3.16 WasPressed()

```
bool WasPressed (
    NetworkButtons previous,
    int button )
```

Checks if the button at the specified index was pressed since the previous state.

Parameters

<i>previous</i>	The previous state of the buttons.
<i>button</i>	The index of the button to check.

Returns

true if the button was pressed; otherwise, false.

6.178.3.17 WasPressed< T >()

```
bool WasPressed< T > (
    NetworkButtons previous,
    T button )
```

Checks if the button of the specified enum type was pressed since the previous state.

Template Parameters

<i>T</i>	The enum type of the button.
----------	------------------------------

Parameters

<i>previous</i>	The previous state of the buttons.
<i>button</i>	The button of type T to check.

Returns

true if the button was pressed; otherwise, false.

Type Constraints

T* : *unmanaged

T* : *Enum

6.178.3.18 WasReleased()

```
bool WasReleased (
    NetworkButtons previous,
    int button )
```

Checks if the button at the specified index was released since the previous state.

Parameters

<i>previous</i>	The previous state of the buttons.
<i>button</i>	The index of the button to check.

Returns

true if the button was released; otherwise, false.

6.178.3.19 WasReleased< T >()

```
bool WasReleased< T > (
    NetworkButtons previous,
    T button )
```

Checks if the button of the specified enum type was released since the previous state.

Template Parameters

<i>T</i>	The enum type of the button.
----------	------------------------------

Parameters

<i>previous</i>	The previous state of the buttons.
<i>button</i>	The button of type T to check.

Returns

true if the button was released; otherwise, false.

Type Constraints

T: *unmanaged*

T: *Enum*

6.178.4 Member Data Documentation

6.178.4.1 NetworkButtons

[NetworkButtons](#)

Gets the buttons that were pressed or released since the previous state.

Parameters

<i>previous</i>	The previous state of the buttons.
-----------------	------------------------------------

Returns

A tuple containing the buttons that were pressed and the buttons that were released.

6.178.5 Property Documentation

6.178.5.1 Bits

```
int Bits [get]
```

Gets the bits representing the state of the buttons.

6.179 NetworkConfiguration Class Reference

Main network configuration class.

Inherits IConfigurationSanityCheck.

Public Types

- enum class [ReliableDataTransfers](#)
Flag for allowed Reliable Data transfer modes.

Public Member Functions

- [NetworkConfiguration Init](#) ()
Initializes and creates a copy of this [NetworkProjectConfig](#).
- void [SanityCheck](#) ()

Public Attributes

- double [ConnectionShutdownTime](#) = 1
Default delay between connection changes status to Shutdown (disconnected/invalid), and it actually being released (freeing all references to that particular connection).
- double [ConnectionTimeout](#) = 10
Max allowed time in seconds that the local peer can run without receiving any update from a remote peer. If a client does not receive any update from the server within this period, it will disconnect itself. If a server does not receive any update from a remote client within this period, it will disconnect that particular client.
- [ReliableDataTransfers ReliableDataTransferModes](#)
Current [ReliableDataTransferModes](#) mode.

Properties

- int [ConnectAttempts](#) [get]
Max number of connection attempts that a Client will run when trying to connect to a remote Server.
- double [ConnectInterval](#) [get]
Interval in seconds between each connection attempt from a Client.
- double [ConnectionDefaultRtt](#) [get]
Default assumed RTT in seconds for new connections (before actual RTT has been determined). The real RTT is calculated over time once the connection is established.
- double [ConnectionPingInterval](#) [get]
Interval in seconds between PING messages sent to a remote connection, in order to keep that connection alive.
- int [SocketRecvBufferSize](#) [get]
Size in Kilobytes of the underlying socket receive buffer.
- int [SocketSendBufferSize](#) [get]
Size in Kilobytes of the underlying socket send buffer.

6.179.1 Detailed Description

Main network configuration class.

6.179.2 Member Enumeration Documentation

6.179.2.1 ReliableDataTransfers

enum [ReliableDataTransfers](#) [strong]

Flag for allowed Reliable Data transfer modes.

Enumerator

ClientToServer	Allow Client to Server.
ClientToClientWithServerProxy	Allow Client to Client using Server as Proxy.

6.179.3 Member Function Documentation

6.179.3.1 Init()

```
NetworkConfiguration Init ( )
```

Initializes and creates a copy of this [NetworkProjectConfig](#).

Returns

A copy of this [NetworkProjectConfig](#).

6.179.4 Member Data Documentation

6.179.4.1 ConnectionShutdownTime

```
double ConnectionShutdownTime = 1
```

Default delay between connection changes status to Shutdown (disconnected/invalid), and it actually being released (freeing all references to that particular connection).

6.179.4.2 ConnectionTimeout

```
double ConnectionTimeout = 10
```

Max allowed time in seconds that the local peer can run without receiving any update from a remote peer. If a client does not receive any update from the server within this period, it will disconnect itself. If a server does not receive any update from a remote client within this period, it will disconnect that particular client.

6.179.4.3 ReliableDataTransferModes

`ReliableDataTransfers` `ReliableDataTransferModes`

Initial value:

```
=  
    ReliableDataTransfers.ClientToServer |  
    ReliableDataTransfers.ClientToClientWithServerProxy
```

Current `ReliableDataTransferModes` mode.

6.179.5 Property Documentation

6.179.5.1 ConnectAttempts

```
int ConnectAttempts [get]
```

Max number of connection attempts that a Client will run when trying to connect to a remote Server.

6.179.5.2 ConnectInterval

```
double ConnectInterval [get]
```

Interval in seconds between each connection attempt from a Client.

6.179.5.3 ConnectionDefaultRtt

```
double ConnectionDefaultRtt [get]
```

Default assumed RTT in seconds for new connections (before actual RTT has been determined). The real RTT is calculated over time once the connection is established.

6.179.5.4 ConnectionPingInterval

```
double ConnectionPingInterval [get]
```

Interval in seconds between PING messages sent to a remote connection, in order to keep that connection alive.

Currently unused.

6.179.5.5 SocketRecvBufferSize

```
int SocketRecvBufferSize [get]
```

Size in Kilobytes of the underlying socket receive buffer.

6.179.5.6 SocketSendBufferSize

```
int SocketSendBufferSize [get]
```

Size in Kilobytes of the underlying socket send buffer.

6.180 NetworkDelegates Class Reference

Network Runner Callbacks Delegates

Inherits [INetworkRunnerCallbacks](#).

Public Attributes

- Action< [NetworkRunner](#) > [OnConnectedToServer](#)
- Action< [NetworkRunner](#), [NetAddress](#), [NetConnectFailedReason](#) > [OnConnectFailed](#)
- Action< [NetworkRunner](#), [NetworkRunnerCallbackArgs.ConnectRequest](#), [byte\[\]](#)> [OnConnectRequest](#)
- Action< [NetworkRunner](#), Dictionary< string, object > > [OnCustomAuthenticationResponse](#)
- Action< [NetworkRunner](#), [NetDisconnectReason](#) > [OnDisconnectedFromServer](#)
- Action< [NetworkRunner](#), [HostMigrationToken](#) > [OnHostMigration](#)
- Action< [NetworkRunner](#), [NetworkInput](#) > [OnInput](#)
- Action< [NetworkRunner](#), [PlayerRef](#), [NetworkInput](#) > [OnInputMissing](#)
- Action< [NetworkRunner](#), [NetworkObject](#), [PlayerRef](#) > [OnObjectEnterAOI](#)
- Action< [NetworkRunner](#), [NetworkObject](#), [PlayerRef](#) > [OnObjectExitAOI](#)
- Action< [NetworkRunner](#), [PlayerRef](#) > [OnPlayerJoined](#)
- Action< [NetworkRunner](#), [PlayerRef](#) > [OnPlayerLeft](#)
- Action< [NetworkRunner](#), [PlayerRef](#), [ReliableKey](#), float > [OnReliableDataProgress](#)
- Action< [NetworkRunner](#), [PlayerRef](#), [ReliableKey](#), [ArraySegment< byte >](#) > [OnReliableDataReceived](#)
- Action< [NetworkRunner](#) > [OnSceneLoadDone](#)
- Action< [NetworkRunner](#) > [OnSceneLoadStart](#)
- Action< [NetworkRunner](#), List< [SessionInfo](#) > > [OnSessionListUpdated](#)
- Action< [NetworkRunner](#), [ShutdownReason](#) > [OnShutdown](#)
- Action< [NetworkRunner](#), [SimulationMessagePtr](#) > [OnUserSimulationMessage](#)

Additional Inherited Members

6.180.1 Detailed Description

Network Runner Callbacks Delegates

6.181 NetworkDeserializeMethodAttribute Class Reference

Network Deserialize Method Attribute

Inherits Attribute.

6.181.1 Detailed Description

Network Deserialize Method Attribute

6.182 NetworkDictionary< K, V > Struct Template Reference

[Fusion](#) type for networking Dictionaries. Maximum capacity is fixed, and is set with the [CapacityAttribute](#).

Inherits IEnumerable< KeyValuePair< K, V >>, and [INetworkDictionary](#).

Classes

- struct [Enumerator](#)
Enumerator for NetworkDictionary.

Public Member Functions

- bool [Add](#) (K key, V value)
Adds a new key value pair to the Dictionary. If the key already exists, will return false.
- void INetworkDictionary. [Add](#) (object item)
Adds an item to the networked dictionary.
- void [Clear](#) ()
Remove all entries from the Dictionary, and clear backing memory.
- void [ClrEntry](#) (int entry)
- bool [ContainsKey](#) (K key)
Returns true if the Dictionary contains an entry for the given key.
- bool [ContainsValue](#) (V value, IEqualityComparer< V > equalityComparer=null)
Returns true if the Dictionary contains an entry value which compares as equal to given value.
- int [Find](#) (K key)
- V [Get](#) (K key)
Returns the value for the given key. Will throw an error if the key is not found.
- uint [GetBucketFromHashCode](#) (int hash)
- [Enumerator](#) [GetEnumerator](#) ()
Returns an enumerator that iterates through the NetworkDictionary.
- IEnumerable< KeyValuePair< K, V >> IEnumerable< KeyValuePair< K, V >>. [GetEnumerator](#) ()
- IEnumerable IEnumerable. [GetEnumerator](#) ()
- K [GetKey](#) (int entry)
- int [GetKeyHashCode](#) (K key)
- int [GetNxt](#) (int entry)

- V **GetVal** (int entry)
- int **Insert** (K key, V val)
- [NetworkDictionary](#) (int *data, int capacity, [IElementReaderWriter](#)< K > keyReaderWriter, [IElementReaderWriter](#)< V > valReaderWriter)

Initializes a new instance of the [NetworkDictionary](#) struct with the specified data, capacity, and reader/writers.
- bool **Remove** (K key)

Remove entry from Dictionary.
- bool **Remove** (K key, out V value)

Removes entry from Dictionary. If successful (key existed), returns true and the value of removed item.
- V **Set** (K key, V value)

Sets the value for the given key. Will add a new key if the key does not already exist.
- void **SetKey** (int entry, K key)
- void **SetNxt** (int entry, int next)
- void **SetVal** (int entry, V val)
- [NetworkDictionaryReadOnly](#)< K, V > **ToReadOnly** ()

Converts the current [NetworkDictionary](#) to a read-only version.
- bool **TryGet** (K key, out V value)

Attempts to get the value for a given key. If found, returns true.

Static Public Member Functions

- static implicit [operator NetworkDictionaryReadOnly](#)< K, V > ([NetworkDictionary](#)< K, V > value)

Converts the current [NetworkDictionary](#) to a read-only version.

Public Attributes

- int **_bucketsOffset**
- int **_capacity**
- int * **_data**
- int **_entriesOffset**
- int **_entryStride**
- EqualityComparer< K > **_equalityComparer**
- int **_keyOffset**
- [IElementReaderWriter](#)< K > **_keyReaderWriter**
- int **_nxtOffset**
- int **_valOffset**
- [IElementReaderWriter](#)< V > **_valReaderWriter**

Static Public Attributes

- const int **FREE_COUNT_OFFSET** = 1
- const int **FREE_OFFSET** = 0
- const int **INVALID_ENTRY** = 0
- const int **META_WORD_COUNT** = 3

Meta word count for [NetworkDictionary](#).
- const int **USED_COUNT_OFFSET** = 2

Properties

- `int_free` [get, set]
- `int_freeCount` [get, set]
- `int_usedCount` [get, set]
- `int Capacity` [get]

The maximum number of entries this dictionary may contain.
- `int Count` [get]

Current number of key/value entries in the Dictionary.
- `V this[K key]` [get, set]

Key indexer. Gets/Sets value for specified key.

6.182.1 Detailed Description

`Fusion` type for networking Dictionaries. Maximum capacity is fixed, and is set with the `CapacityAttribute`.

Typical Usage: `[Networked, Capacity(10)]`

```
NetworkDictionary<int, float> syncedDict => default;
```

Usage for modifying data: `var dict = syncedDict; dict.Add(5, 123); dict[5] = 456; dict.Remove(5);`

Template Parameters

<i>K</i>	Key can be a primitive, or an INetworkStruct .
<i>V</i>	Value can be a primitive, or an INetworkStruct .

6.182.2 Constructor & Destructor Documentation

6.182.2.1 NetworkDictionary()

```
NetworkDictionary (
    int * data,
    int capacity,
    IElementReaderWriter< K > keyReaderWriter,
    IElementReaderWriter< V > valReaderWriter )
```

Initializes a new instance of the `NetworkDictionary` struct with the specified data, capacity, and reader/writers.

Parameters

<i>data</i>	The pointer to the data of the dictionary.
<i>capacity</i>	The capacity of the dictionary.
<i>keyReaderWriter</i>	The reader/writer for the keys of the dictionary.
<i>valReaderWriter</i>	The reader/writer for the values of the dictionary.

6.182.3 Member Function Documentation

6.182.3.1 Add() [1/2]

```
bool Add (
    K key,
    V value )
```

Adds a new key value pair to the Dictionary. If the key already exists, will return false.

6.182.3.2 Add() [2/2]

```
void INetworkDictionary. Add (
    object item )
```

Adds an item to the networked dictionary.

Parameters

<i>item</i>	The item to add to the dictionary.
-------------	------------------------------------

Implements [INetworkDictionary](#).

6.182.3.3 Clear()

```
void Clear ( )
```

Remove all entries from the Dictionary, and clear backing memory.

6.182.3.4 ContainsKey()

```
bool ContainsKey (
    K key )
```

Returns true if the Dictionary contains an entry for the given key.

6.182.3.5 ContainsValue()

```
bool ContainsValue (
    V value,
    IEqualityComparer< V > equalityComparer = null )
```

Returns true if the Dictionary contains an entry value which compares as equal to given value.

Parameters

<i>value</i>	The value to compare against.
<i>equalityComparer</i>	Specify custom IEqualityComparer to be used for compare.

6.182.3.6 Get()

```
V Get (
    K key )
```

Returns the value for the given key. Will throw an error if the key is not found.

6.182.3.7 GetEnumerator()

```
IEnumerator GetEnumerator ( )
```

Returns an enumerator that iterates through the [NetworkDictionary](#).

6.182.3.8 operator NetworkDictionaryReadOnly< K, V >()

```
static implicit operator NetworkDictionaryReadOnly< K, V > (
    NetworkDictionary< K, V > value ) [static]
```

Converts the current [NetworkDictionary](#) to a read-only version.

Parameters

<i>value</i>	The NetworkDictionary to convert.
--------------	---

Returns

A new instance of [NetworkDictionaryReadOnly](#) with the same data, capacity, and reader/writers as the current [NetworkDictionary](#).

6.182.3.9 Remove() [1/2]

```
bool Remove (
    K key )
```

Remove entry from Dictionary.

Parameters

<i>key</i>	The key to remove.
------------	--------------------

Returns

Returns true if key was found.

6.182.3.10 Remove() [2/2]

```
bool Remove (
    K key,
    out V value )
```

Removes entry from Dictionary. If successful (key existed), returns true and the value of removed item.

Parameters

<i>key</i>	The key to remove.
<i>value</i>	Returns value of removed item. Returns default value if key did not exist.

Returns

Returns true if key was found.

6.182.3.11 Set()

```
V Set (
    K key,
    V value )
```

Sets the value for the given key. Will add a new key if the key does not already exist.

6.182.3.12 ToReadOnly()

```
NetworkDictionaryReadOnly<K, V> ToReadOnly ( )
```

Converts the current [NetworkDictionary](#) to a read-only version.

Returns

A new instance of [NetworkDictionaryReadOnly](#) with the same data, capacity, and reader/writers as the current [NetworkDictionary](#).

6.182.3.13 TryGet()

```
bool TryGet (
    K key,
    out V value )
```

Attempts to get the value for a given key. If found, returns true.

Parameters

<i>key</i>	The key to remove.
<i>value</i>	Returns value of removed item. Returns default value if key did not exist.

Returns

Returns true if key was found.

6.182.4 Member Data Documentation

6.182.4.1 META_WORD_COUNT

```
const int META_WORD_COUNT = 3 [static]
```

Meta word count for [NetworkDictionary](#).

6.182.5 Property Documentation

6.182.5.1 Capacity

```
int Capacity [get]
```

The maximum number of entries this dictionary may contain.

6.182.5.2 Count

```
int Count [get]
```

Current number of key/value entries in the Dictionary.

6.182.5.3 this[K key]

V this[K key] [get], [set]

Key indexer. Gets/Sets value for specified key.

6.183 NetworkDictionary< K, V >.Enumerator Struct Reference

[Enumerator](#) for [NetworkDictionary](#).

Inherits [IEnumerator< KeyValuePair< K, V >>](#).

Public Member Functions

- void [Dispose](#) ()
Dispose enumerator.
- bool [MoveNext](#) ()
Move to next entry in dictionary.
- void [Reset](#) ()
Reset enumerator.

Public Attributes

- int [_bucket](#)
- [NetworkDictionary< K, V > _dict](#)
- int [_entry](#)

Properties

- [KeyValuePair< K, V > Current](#) [get]
Current key/value pair.
- object [IEnumerator](#). [Current](#) [get]

6.183.1 Detailed Description

[Enumerator](#) for [NetworkDictionary](#).

6.183.2 Member Function Documentation

6.183.2.1 Dispose()

```
void Dispose ( )
```

Dispose enumerator.

6.183.2.2 MoveNext()

```
bool MoveNext ( )
```

Move to next entry in dictionary.

Returns

Returns true if there is a next entry.

6.183.2.3 Reset()

```
void Reset ( )
```

Reset enumerator.

6.183.3 Property Documentation

6.183.3.1 Current

```
KeyValuePair<K, V> Current [get]
```

Current key/value pair.

Exceptions

<i>InvalidOperationException</i>	Thrown if enumerator is not valid.
----------------------------------	------------------------------------

6.184 NetworkDictionaryReadOnly< K, V > Struct Template Reference

A read-only version of NetworkDictionary<TKey,TValue>.

Public Member Functions

- int **Find** (K key)
- V **Get** (K key)

Returns the value for the given key. Will throw an error if the key is not found.
- uint **GetBucketFromHashCode** (int hash)
- K **GetKey** (int entry)
- int **GetNxt** (int entry)
- V **GetVal** (int entry)
- bool **TryGet** (K key, out V value)

Attempts to get the value for a given key. If found, returns true.

Public Attributes

- readonly int **_bucketsOffset**
- readonly int **_capacity**
- readonly int * **_data**
- readonly int **_entriesOffset**
- readonly int **_entryStride**
- readonly EqualityComparer< K > **_equalityComparer**
- readonly int **_keyOffset**
- readonly [IElementReaderWriter](#)< K > **_keyReaderWriter**
- readonly int **_nxtOffset**
- readonly int **_valOffset**
- readonly [IElementReaderWriter](#)< V > **_valReaderWriter**

Static Public Attributes

- const int **FREE_COUNT_OFFSET** = 1
- const int **FREE_OFFSET** = 0
- const int **INVALID_ENTRY** = 0
- const int **USED_COUNT_OFFSET** = 2

Properties

- int **_free** [get]
- int **_freeCount** [get]
- int **_usedCount** [get]
- int **Capacity** [get]

The maximum number of entries this dictionary may contain.
- int **Count** [get]

Current number of key/value entries in the Dictionary.

6.184.1 Detailed Description

A read-only version of `NetworkDictionary<TKey,TValue>`.

Template Parameters

<i>K</i>	The type of the key.
<i>V</i>	The type of the value.

6.184.2 Member Function Documentation

6.184.2.1 Get()

```
V Get (
    K key )
```

Returns the value for the given key. Will throw an error if the key is not found.

6.184.2.2 TryGet()

```
bool TryGet (
    K key,
    out V value )
```

Attempts to get the value for a given key. If found, returns true.

Parameters

<i>key</i>	The key to remove.
<i>value</i>	Returns value of removed item. Returns default value if key did not exist.

Returns

Returns true if key was found.

6.184.3 Property Documentation

6.184.3.1 Capacity

```
int Capacity [get]
```

The maximum number of entries this dictionary may contain.

6.184.3.2 Count

`int Count [get]`

Current number of key/value entries in the Dictionary.

6.185 NetworkedAttribute Class Reference

Inherits Attribute.

Public Member Functions

- [NetworkedAttribute \(\)](#)
Default constructor for [NetworkedAttribute](#)

Properties

- string [Default](#) [get, set]
Name of the field that holds the default value for this networked property.

6.185.1 Detailed Description

Flags a property of [NetworkBehaviour](#) for network state synchronization. The property should have empty get and set defines, which will automatically be replaced with networking code via IL Weaving.

Inside of [INetworkStruct](#), do not use AutoProperties (get; set;), as these will introduce managed types into the struct, which are not allowed. Instead use '`=> default`'. | [[Networked](#)]
| `public string StringProp { get => default; set { } }`

6.185.2 Constructor & Destructor Documentation

6.185.2.1 NetworkedAttribute()

`NetworkedAttribute ()`

Default constructor for [NetworkedAttribute](#)

6.185.3 Property Documentation

6.185.3.1 Default

```
string Default [get], [set]
```

Name of the field that holds the default value for this networked property.

6.186 NetworkedWeavedArrayAttribute Class Reference

Attribute applied to an array property by the weaver.

Inherits Attribute.

Public Member Functions

- [NetworkedWeavedArrayAttribute](#) (int capacity, int elementWordCount, Type elementReaderWriter)

Properties

- int [Capacity](#) [get]
Array capacity.
- Type [ElementReaderWriterType](#) [get]
ReaderWriter type for the element.
- int [ElementWordCount](#) [get]
Word count of each element.

6.186.1 Detailed Description

Attribute applied to an array property by the weaver.

6.186.2 Property Documentation

6.186.2.1 Capacity

```
int Capacity [get]
```

Array capacity.

6.186.2.2 ElementReaderWriterType

Type `ElementReaderWriterType` [get]

ReaderWriter type for the element.

6.186.2.3 ElementWordCount

int `ElementWordCount` [get]

Word count of each element.

6.187 NetworkedWeavedAttribute Class Reference

Networked Weaved Attribute

Inherits Attribute.

Public Member Functions

- [NetworkedWeavedAttribute](#) (int wordOffset, int wordCount)
NetworkedWeavedAttribute Constructor

Properties

- int [WordCount](#) [get]
Networked Property Word Count
- int [WordOffset](#) [get]
Networked Property Word Offset

6.187.1 Detailed Description

Networked Weaved Attribute

Networked Property Attribute

6.187.2 Constructor & Destructor Documentation

6.187.2.1 NetworkedWeavedAttribute()

```
NetworkedWeavedAttribute (  
    int wordOffset,  
    int wordCount )
```

[NetworkedWeavedAttribute](#) Constructor

Parameters

<code>wordOffset</code>	WordOffset
<code>wordCount</code>	WordCount

6.187.3 Property Documentation

6.187.3.1 WordCount

```
int WordCount [get]
```

Networked Property Word Count

6.187.3.2 WordOffset

```
int WordOffset [get]
```

Networked Property Word Offset

6.188 NetworkedWeavedDictionaryAttribute Class Reference

Attribute applied to a dictionary property by the weaver.

Inherits Attribute.

Public Member Functions

- [NetworkedWeavedDictionaryAttribute](#) (int capacity, int keyWordCount, int elementWordCount, Type keyReaderWriterType, Type valueReaderWriterType)

Properties

- int [Capacity](#) [get]
Capacity of the dictionary.
- Type [KeyReaderWriterType](#) [get, set]
- int [KeyWordCount](#) [get]
Word count for the key.
- Type [ValueReaderWriterType](#) [get, set]
- int [ValueWordCount](#) [get]
Word count for the value.

6.188.1 Detailed Description

Attribute applied to a dictionary property by the weaver.

6.188.2 Property Documentation

6.188.2.1 Capacity

```
int Capacity [get]
```

Capacity of the dictionary.

6.188.2.2 KeyWordCount

```
int KeyWordCount [get]
```

Word count for the key.

6.188.2.3 ValueWordCount

```
int ValueWordCount [get]
```

Word count for the value.

6.189 NetworkedWeavedLinkedListAttribute Class Reference

Attribute applied to a list property by the weaver.

Inherits Attribute.

Public Member Functions

- [NetworkedWeavedLinkedListAttribute](#) (int capacity, int elementWordCount, Type elementReaderWriterType)

Properties

- int [Capacity](#) [get]
List capacity.
- Type [ElementReaderWriterType](#) [get]
ReaderWriter type for the element.
- int [ElementWordCount](#) [get]
Word count of each element.

6.189.1 Detailed Description

Attribute applied to a list property by the weaver.

6.189.2 Property Documentation

6.189.2.1 Capacity

```
int Capacity [get]
```

List capacity.

6.189.2.2 ElementReaderWriterType

```
Type ElementReaderWriterType [get]
```

ReaderWriter type for the element.

6.189.2.3 ElementWordCount

```
int ElementWordCount [get]
```

Word count of each element.

6.190 NetworkEvents Class Reference

Companion component for [NetworkRunner](#). Exposes [INetworkRunnerCallbacks](#) as UnityEvents, which can be wired up to other components in the inspector.

Inherits [Behaviour](#), and [INetworkRunnerCallbacks](#).

Classes

- class [ConnectFailedEvent](#)
UnityEvent for ConnectFailed
- class [ConnectRequestEvent](#)
UnityEvent for ConnectRequest
- class [CustomAuthenticationResponse](#)
UnityEvent for Custom Authentication
- class [DisconnectFromServerEvent](#)
UnityEvent for DisconnectFromServer
- class [HostMigrationEvent](#)
UnityEvent for HostMigration
- class [InputEvent](#)
UnityEvent for NetworkInput
- class [InputPlayerEvent](#)
UnityEvent for NetworkInput with PlayerRef
- class [ObjectEvent](#)
UnityEvent for NetworkObject
- class [ObjectPlayerEvent](#)
UnityEvent for NetworkObject with PlayerRef
- class [PlayerEvent](#)
UnityEvent for PlayerRef
- class [ReliableDataEvent](#)
UnityEvent for Reliable Data
- class [ReliableProgressEvent](#)
UnityEvent for Reliable Data Progress
- class [RunnerEvent](#)
UnityEvent for NetworkRunner
- class [SessionListUpdateEvent](#)
UnityEvent for SessionInfo List
- class [ShutdownEvent](#)
UnityEvent for Shutdown
- class [SimulationMessageEvent](#)
UnityEvent for SimulationMessage

Public Attributes

- [RunnerEvent](#) OnConnectedToServer
- [ConnectFailedEvent](#) OnConnectFailed
- [ConnectRequestEvent](#) OnConnectRequest
- [CustomAuthenticationResponse](#) OnCustomAuthenticationResponse
- [DisconnectFromServerEvent](#) OnDisconnectedFromServer
- [HostMigrationEvent](#) OnHostMigration
- [InputEvent](#) OnInput
- [InputPlayerEvent](#) OnInputMissing
- [ObjectPlayerEvent](#) OnObjectEnterAOI
- [ObjectPlayerEvent](#) OnObjectExitAOI
- [ReliableDataEvent](#) OnReliableData
- [ReliableProgressEvent](#) OnReliableProgress
- [RunnerEvent](#) OnSceneLoadDone
- [RunnerEvent](#) OnSceneLoadStart
- [SessionListUpdateEvent](#) OnSessionListUpdate
- [ShutdownEvent](#) OnShutdown
- [SimulationMessageEvent](#) OnSimulationMessage
- [PlayerEvent](#) PlayerJoined
- [PlayerEvent](#) PlayerLeft

Additional Inherited Members

6.190.1 Detailed Description

Companion component for [NetworkRunner](#). Exposes [INetworkRunnerCallbacks](#) as UnityEvents, which can be wired up to other components in the inspector.

6.191 NetworkEvents.ConnectFailedEvent Class Reference

UnityEvent for ConnectFailed

Inherits UnityEvent< NetworkRunner, NetAddress, NetConnectFailedReason >.

6.191.1 Detailed Description

UnityEvent for ConnectFailed

6.192 NetworkEvents.ConnectRequestEvent Class Reference

UnityEvent for ConnectRequest

Inherits UnityEvent< NetworkRunner, NetworkRunnerCallbackArgs.ConnectRequest, byte[] >.

6.192.1 Detailed Description

UnityEvent for ConnectRequest

6.193 NetworkEvents.CustomAuthenticationResponse Class Reference

UnityEvent for Custom Authentication

Inherits UnityEvent< NetworkRunner, Dictionary< string, object >>.

6.193.1 Detailed Description

UnityEvent for Custom Authentication

6.194 NetworkEvents.DisconnectFromServerEvent Class Reference

UnityEvent for DisconnectFromServer

Inherits UnityEvent< NetworkRunner, NetDisconnectReason >.

6.194.1 Detailed Description

UnityEvent for DisconnectFromServer

6.195 NetworkEvents.HostMigrationEvent Class Reference

UnityEvent for HostMigration

Inherits UnityEvent< NetworkRunner, HostMigrationToken >.

6.195.1 Detailed Description

UnityEvent for HostMigration

6.196 NetworkEvents.InputEvent Class Reference

UnityEvent for [NetworkInput](#)

Inherits UnityEvent< NetworkRunner, NetworkInput >.

6.196.1 Detailed Description

UnityEvent for [NetworkInput](#)

6.197 NetworkEvents.InputPlayerEvent Class Reference

UnityEvent for [NetworkInput](#) with [PlayerRef](#)

Inherits UnityEvent< NetworkRunner, PlayerRef, NetworkInput >.

6.197.1 Detailed Description

UnityEvent for [NetworkInput](#) with [PlayerRef](#)

6.198 NetworkEvents.ObjectEvent Class Reference

UnityEvent for [NetworkObject](#)

Inherits UnityEvent< NetworkRunner, NetworkObject >.

6.198.1 Detailed Description

UnityEvent for [NetworkObject](#)

6.199 NetworkEvents.ObjectPlayerEvent Class Reference

UnityEvent for [NetworkObject](#) with [PlayerRef](#)

Inherits UnityEvent< [NetworkRunner](#), [NetworkObject](#), [PlayerRef](#) >.

6.199.1 Detailed Description

UnityEvent for [NetworkObject](#) with [PlayerRef](#)

6.200 NetworkEvents.PlayerEvent Class Reference

UnityEvent for [PlayerRef](#)

Inherits UnityEvent< [NetworkRunner](#), [PlayerRef](#) >.

6.200.1 Detailed Description

UnityEvent for [PlayerRef](#)

6.201 NetworkEvents.ReliableDataEvent Class Reference

UnityEvent for Reliable Data

Inherits UnityEvent< [NetworkRunner](#), [PlayerRef](#), [ReliableKey](#), [ArraySegment< byte >>](#) >.

6.201.1 Detailed Description

UnityEvent for Reliable Data

6.202 NetworkEvents.ReliableProgressEvent Class Reference

UnityEvent for Reliable Data Progress

Inherits UnityEvent< [NetworkRunner](#), [PlayerRef](#), [ReliableKey](#), [float](#) >.

6.202.1 Detailed Description

UnityEvent for Reliable Data Progress

6.203 NetworkEvents.RunnerEvent Class Reference

UnityEvent for [NetworkRunner](#)

Inherits UnityEvent< NetworkRunner >.

6.203.1 Detailed Description

UnityEvent for [NetworkRunner](#)

6.204 NetworkEvents.SessionListUpdateEvent Class Reference

UnityEvent for [SessionInfo](#) List

Inherits UnityEvent< NetworkRunner, List< SessionInfo >>.

6.204.1 Detailed Description

UnityEvent for [SessionInfo](#) List

6.205 NetworkEvents.ShutdownEvent Class Reference

UnityEvent for Shutdown

Inherits UnityEvent< NetworkRunner, ShutdownReason >.

6.205.1 Detailed Description

UnityEvent for Shutdown

6.206 NetworkEvents.SimulationMessageEvent Class Reference

UnityEvent for [SimulationMessage](#)

Inherits UnityEvent< NetworkRunner, SimulationMessagePtr >.

6.206.1 Detailed Description

UnityEvent for [SimulationMessage](#)

6.207 NetworkId Struct Reference

The unique identifier for a network entity.

Inherits [INetworkStruct](#), [IEquatable< NetworkId >](#), [IComparable](#), and [IComparable< NetworkId >](#).

Classes

- class [EqualityComparer](#)
IEqualityComparer interface for [NetworkId](#) objects.

Public Member Functions

- int [CompareTo](#) ([NetworkId](#) other)
Compares the current [NetworkId](#) object with another [NetworkId](#) object.
- int [IComparable.CompareTo](#) (object obj)
- bool [Equals](#) ([NetworkId](#) other)
Determines whether the current [NetworkId](#) object is equal to another [NetworkId](#) object.
- override bool [Equals](#) (object obj)
Determines whether the specified object is equal to the current [NetworkId](#) object.
- override int [GetHashCode](#) ()
Get the hash code for this [NetworkId](#).
- string [ToNamePrefixString](#) ()
String conversion specifically for use in prefixing names of [GameObjects](#).
- override string [ToString](#) ()
String representation of the [NetworkId](#).
- void [Write](#) ([NetBitBuffer](#) *buffer)
Writes this [NetworkId](#) to the provided [NetBitBuffer](#).

Static Public Member Functions

- static implicit operator bool ([NetworkId](#) id)
Converts the [NetworkId](#) object to a boolean value.
- static bool operator!= ([NetworkId](#) a, [NetworkId](#) b)
Determines whether two [NetworkId](#) objects are not equal.
- static bool operator== ([NetworkId](#) a, [NetworkId](#) b)
Determines whether two [NetworkId](#) objects are equal.
- static [NetworkId](#) Read ([NetBitBuffer](#) *buffer)
Reads a [NetworkId](#) from the provided [NetBitBuffer](#).
- static void Write ([NetBitBuffer](#) *buffer, [NetworkId](#) id)
Writes the [NetworkId](#) to the provided [NetBitBuffer](#).

Public Attributes

- uint [Raw](#)

The raw value of the network id.

Static Public Attributes

- const int [ALIGNMENT](#) = 4
The alignment of the network id in bytes.
- const int [BLOCK_SIZE](#) = 8
The size of the network id block in bytes.
- const uint [RAW_PHYSICS_INFO](#) = 4u
- const uint [RAW_PLAYER_REF_DATA_ARRAY](#) = 2u
- const uint [RAW_RUNTIME_CONFIG](#) = 1u
- const uint [RAW_SCENE_INFO](#) = 3u
- const int [SIZE](#) = 4

The size of the network id in bytes.

Properties

- static [EqualityComparer Comparer](#) = new [EqualityComparer\(\)](#) [get]
The IEqualityComparer for [NetworkId](#) objects.
- bool [IsReserved](#) [get]
Signal if the network id is reserved.
- bool [IsValid](#) [get]
Signal if the network id is valid.

6.207.1 Detailed Description

The unique identifier for a network entity.

6.207.2 Member Function Documentation

6.207.2.1 CompareTo()

```
int CompareTo (  
    NetworkId other )
```

Compares the current [NetworkId](#) object with another [NetworkId](#) object.

Parameters

<i>other</i>	A NetworkId object to compare with this object.
--------------	---

Returns

A value that indicates the relative order of the objects being compared.

6.207.2.2 Equals() [1/2]

```
bool Equals (  
    NetworkId other )
```

Determines whether the current [NetworkId](#) object is equal to another [NetworkId](#) object.

Parameters

<i>other</i>	A NetworkId object to compare with this object.
--------------	---

Returns

true if the current object is equal to the other parameter; otherwise, false.

6.207.2.3 Equals() [2/2]

```
override bool Equals (  
    object obj )
```

Determines whether the specified object is equal to the current [NetworkId](#) object.

Parameters

<i>obj</i>	The object to compare with the current object.
------------	--

Returns

true if the specified object is equal to the current object; otherwise, false.

6.207.2.4 GetHashCode()

```
override int GetHashCode ( )
```

Get the hash code for this [NetworkId](#).

6.207.2.5 operator bool()

```
static implicit operator bool (
    NetworkId id ) [static]
```

Converts the [NetworkId](#) object to a boolean value.

6.207.2.6 operator"!="()

```
static bool operator!= (
    NetworkId a,
    NetworkId b ) [static]
```

Determines whether two [NetworkId](#) objects are not equal.

6.207.2.7 operator==()

```
static bool operator== (
    NetworkId a,
    NetworkId b ) [static]
```

Determines whether two [NetworkId](#) objects are equal.

6.207.2.8 Read()

```
static NetworkId Read (
    NetBitBuffer * buffer ) [static]
```

Reads a [NetworkId](#) from the provided [NetBitBuffer](#).

Parameters

<i>buffer</i>	The buffer to read the NetworkId from.
---------------	--

Returns

The [NetworkId](#) read from the buffer.

6.207.2.9 ToNamePrefixString()

```
string ToNamePrefixString ( )
```

String conversion specifically for use in prefixing names of GameObjects.

6.207.2.10 ToString()

```
override string ToString ( )
```

String representation of the [NetworkId](#).

6.207.2.11 Write() [1/2]

```
void Write (
    NetBitBuffer * buffer )
```

Writes this [NetworkId](#) to the provided NetBitBuffer.

Parameters

<i>buffer</i>	The buffer to write this NetworkId to.
---------------	--

6.207.2.12 Write() [2/2]

```
static void Write (
    NetBitBuffer * buffer,
    NetworkId id ) [static]
```

Writes the [NetworkId](#) to the provided NetBitBuffer.

Parameters

<i>buffer</i>	The buffer to write the NetworkId to.
<i>id</i>	The NetworkId to write.

6.207.3 Member Data Documentation

6.207.3.1 ALIGNMENT

```
const int ALIGNMENT = 4 [static]
```

The alignment of the network id in bytes.

6.207.3.2 BLOCK_SIZE

```
const int BLOCK_SIZE = 8 [static]
```

The size of the network id block in bytes.

6.207.3.3 Raw

```
uint Raw
```

The raw value of the network id.

6.207.3.4 SIZE

```
const int SIZE = 4 [static]
```

The size of the network id in bytes.

6.207.4 Property Documentation

6.207.4.1 Comparer

```
EqualityComparer Comparer = new EqualityComparer() [static], [get]
```

The IEqualityComparer for [NetworkId](#) objects.

6.207.4.2 IsReserved

```
bool IsReserved [get]
```

Signal if the network id is reserved.

6.207.4.3 IsValid

```
bool IsValid [get]
```

Signal if the network id is valid.

6.208 NetworkId.EqualityComparer Class Reference

IEqualityComparer interface for [NetworkId](#) objects.

Inherits IEqualityComparer< [NetworkId](#) >.

Public Member Functions

- bool [Equals](#) ([NetworkId](#) a, [NetworkId](#) b)
Determines whether the specified [NetworkId](#) objects are equal.
- int [GetHashCode](#) ([NetworkId](#) id)
Returns a hash code for the specified [NetworkId](#) object.

6.208.1 Detailed Description

IEqualityComparer interface for [NetworkId](#) objects.

6.208.2 Member Function Documentation

6.208.2.1 Equals()

```
bool Equals (  
    NetworkId a,  
    NetworkId b )
```

Determines whether the specified [NetworkId](#) objects are equal.

6.208.2.2 GetHashCode()

```
int GetHashCode (  
    NetworkId id )
```

Returns a hash code for the specified [NetworkId](#) object.

6.209 NetworkInput Struct Reference

[NetworkInput](#) Struct

Public Member Functions

- bool [Convert](#) (Type type)
- bool [Convert](#)< T > ()
Converts the Type of this [INetworkInput](#) to another type
- T [Get](#)< T > ()
Gets the content of this [INetworkInput](#) as another type
- bool [Is](#)< T > ()
Checks if this [INetworkInput](#) is of a certain type
- bool [Set](#)< T > (T value)
Sets the content of this [INetworkInput](#) to another type
- bool [TryGet](#)< T > (out T input)
Tries to export data as the indicated T [INetworkInput](#) struct.
- bool [TrySet](#)< T > (T input)
Tries to import data from a [INetworkInput](#) struct.

Properties

- uint * [Data](#) [get]
Data pointer of the [NetworkInput](#)
- bool [IsValid](#) [get]
Signal if the [NetworkInput](#) is valid or not
- Type [Type](#) [get]
Get the Type associated with this [NetworkInput](#)
- int [WordCount](#) [get]
Number of Words for the [NetworkInput](#)

6.209.1 Detailed Description

[NetworkInput](#) Struct

6.209.2 Member Function Documentation

6.209.2.1 Convert< T >()

```
bool Convert< T > ( )
```

Converts the Type of this [INetworkInput](#) to another type

Type Constraints

T: ***unmanaged***
T: ***INetworkInput***
T: ***Convert***
T: ***typeof***
T: ***T***

6.209.2.2 Get< T >()

```
T Get< T > ( )
```

Gets the content of this [INetworkInput](#) as another type

Type Constraints

T : unmanaged
T : INetworkInput

6.209.2.3 Is< T >()

```
bool Is< T > ( )
```

Checks if this [INetworkInput](#) is of a certain type

Type Constraints

T : unmanaged
T : INetworkInput

6.209.2.4 Set< T >()

```
bool Set< T > (
    T value )
```

Sets the content of this [INetworkInput](#) to another type

Type Constraints

T : unmanaged
T : INetworkInput

6.209.2.5 TryGet< T >()

```
bool TryGet< T > (
    out T input )
```

Tries to export data as the indicated T [INetworkInput](#) struct.

Type Constraints

T : unmanaged
T : INetworkInput

6.209.2.6 TrySet< T >()

```
bool TrySet< T > (
    T input )
```

Tries to import data from a [INetworkInput](#) struct.

Type Constraints

T : *unmanaged*

T : *INetworkInput*

6.209.3 Property Documentation

6.209.3.1 Data

```
uint* Data [get]
```

Data pointer of the [NetworkInput](#)

6.209.3.2 IsValid

```
bool IsValid [get]
```

Signal if the [NetworkInput](#) is valid or not

6.209.3.3 Type

```
Type Type [get]
```

Get the Type associated with this [NetworkInput](#)

6.209.3.4 WordCount

```
int WordCount [get]
```

Number of Words for the [NetworkInput](#)

6.210 NetworkInputUtils Class Reference

Utility methods for [NetworkInput](#)

Static Public Member Functions

- static int [GetMaxWordCount](#) ()
Get the max word count from all registered types
- static Type [GetType](#) (int typeKey)
Get the Type based on its associate Key
- static int [GetTypeKey](#) (Type type)
Get the Key associate with the argument Type
- static int [GetWordCount](#) (Type type)
Get Type Word Count if it is of type [NetworkInput](#)

6.210.1 Detailed Description

Utility methods for [NetworkInput](#)

6.210.2 Member Function Documentation

6.210.2.1 GetMaxWordCount()

```
static int GetMaxWordCount ( ) [static]
```

Get the max word count from all registered types

6.210.2.2 GetType()

```
static Type GetType (  
    int typeKey ) [static]
```

Get the Type based on its associate Key

Parameters

<i>typeKey</i>	Key associated with a Type
----------------	----------------------------

Returns

Type associated with the Key, or null otherwise

6.210.2.3 GetTypeKey()

```
static int GetTypeKey (  
    Type type ) [static]
```

Get the Key associate with the argument Type

Parameters

<i>type</i>	Type to check for the key
-------------	---------------------------

Returns

Associated Type Key, or an exception if not found

6.210.2.4 GetWordCount()

```
static int GetWordCount (  
    Type type ) [static]
```

Get Type Word Count if it is of type [NetworkInput](#)

Parameters

<i>type</i>	Type to check for word count
-------------	------------------------------

Returns

Number of words for the [NetworkInput](#)

6.211 NetworkInputWeavedAttribute Class Reference

Network Input Weaved Attribute

Inherits [Attribute](#).

Public Member Functions

- [NetworkInputWeavedAttribute](#) (int wordCount)
NetworkInputWeavedAttribute Constructor

Properties

- int [WordCount](#) [get]
NetworkInput Word Count

6.211.1 Detailed Description

Network Input Weaved Attribute

6.211.2 Constructor & Destructor Documentation

6.211.2.1 NetworkInputWeavedAttribute()

```
NetworkInputWeavedAttribute (  
    int wordCount )
```

[NetworkInputWeavedAttribute](#) Constructor

Parameters

<i>wordCount</i>	WordCount
------------------	---------------------------

6.211.3 Property Documentation

6.211.3.1 WordCount

```
int WordCount [get]
```

[NetworkInput](#) Word Count

6.212 NetworkLinkedList< T > Struct Template Reference

[Fusion](#) type for networking LinkedLists. Maximum capacity is fixed, and is set with the [CapacityAttribute](#).

Typical Usage:

Inherits [IEnumerable< T >](#), and [INetworkLinkedList](#).

Classes

- struct [Enumerator](#)
Enumerator for [NetworkLinkedList<T>](#).

Public Member Functions

- void [INetworkLinkedList](#). [Add](#) (object item)
Adds an item to the networked linked list.
- void [Add](#) (T value)
Adds a value to the end of the list.
- void [Clear](#) ()
Removes and clears all list elements.
- bool [Contains](#) (T value)
Returns true if the value already exists in the list.
- bool [Contains](#) (T value, [IEqualityComparer< T >](#) comparer)
Returns true if the value already exists in the list.
- int * [Entry](#) (int index)
- int * [FindFreeEntry](#) (out int index)
- T [Get](#) (int index)
Returns the value at supplied index.
- int * [GetEntryByListIndex](#) (int listIndex)
- [Enumerator](#) [GetEnumerator](#) ()
Get the enumerator for the list.
- [IEnumerator< T >](#) [IEnumerable< T >](#). [GetEnumerator](#) ()
- [IEnumerator](#) [IEnumerable](#). [GetEnumerator](#) ()
- int [IndexOf](#) (T value)
Returns the index with this value. Returns -1 if not found.
- int [IndexOf](#) (T value, [IEqualityComparer< T >](#) equalityComparer)
Returns the index of the first occurrence of a value in the [NetworkLinkedList](#).
- [NetworkLinkedList](#) (byte *data, int capacity, [IElementReaderWriter< T >](#) rw)
Initializes a new instance of the [NetworkLinkedList](#) struct with the specified data, capacity, and reader/writer.
- T [Read](#) (int *entry)
- [NetworkLinkedList< T >](#) [Remap](#) (void *list)
Remaps the current [NetworkLinkedList](#) to a new memory location.
- bool [Remove](#) (T value)
Removes the first found element with indicated value.
- bool [Remove](#) (T value, [IEqualityComparer< T >](#) equalityComparer)
Removes the first found element with indicated value.
- void [RemoveEntry](#) (int *entry, int entryIndex)
- T [Set](#) (int index, T value)
Sets the value at supplied index.
- void [Write](#) (int *entry, T value)

Public Attributes

- int [_capacity](#)
- int * [_data](#)
- [IElementReaderWriter< T >](#) [_rw](#)
- int [_stride](#)

Static Public Attributes

- const int **COUNT** = 0
- const int **ELEMENT_WORDS** = 2
Returns the number of words required to store a single element.
- const int **HEAD** = 1
- const int **INVALID** = 0
- const int **META_WORDS** = 3
Returns the number of words required to store the list metadata.
- const int **NEXT** = 1
- const int **OFFSET** = 1
- const int **PREV** = 0
- const int **TAIL** = 2

Properties

- int **Capacity** [get]
Returns the max element count.
- int **Count** [get]
Returns the current element count.
- int **Head** [get, set]
- int **Tail** [get, set]
- T **this[int index]** [get, set]
Element indexer.

6.212.1 Detailed Description

Fusion type for networking LinkedLists. Maximum capacity is fixed, and is set with the [CapacityAttribute](#).

Typical Usage:

```
[Networked, Capacity(10)]
NetworkLinkedList<int> syncedLinkedList => default;
```

Optional usage (for NetworkBehaviours ONLY - this is not legal in INetworkStructs): [Networked, Capacity(4)]

```
NetworkLinkedList<int> syncedLinkedList { get; } = MakeInitializer(new int[]
{ 1, 2, 3, 4 });
```

Usage for modifying data: `var list = syncedLinkedList; list.Add(123); list[0] = 456; list.Remove(0);`

Template Parameters

T	T can be a primitive, or an INetworkStruct .
----------	--

6.212.2 Constructor & Destructor Documentation

6.212.2.1 NetworkLinkedList()

```
NetworkLinkedList (
    byte * data,
    int capacity,
    IElementReaderWriter< T > rw )
```

Initializes a new instance of the [NetworkLinkedList](#) struct with the specified data, capacity, and reader/writer.

Parameters

<i>data</i>	The pointer to the data of the list.
<i>capacity</i>	The capacity of the list.
<i>rw</i>	The reader/writer for the elements of the list.

6.212.3 Member Function Documentation

6.212.3.1 Add() [1/2]

```
void INetworkLinkedList. Add (
    object item )
```

Adds an item to the networked linked list.

Parameters

<i>item</i>	The item to add to the linked list.
-------------	-------------------------------------

Implements [INetworkLinkedList](#).

6.212.3.2 Add() [2/2]

```
void Add (
    T value )
```

Adds a value to the end of the list.

Parameters

<i>value</i>	Value to add.
--------------	---------------

6.212.3.3 Clear()

```
void Clear ( )
```

Removes and clears all list elements.

6.212.3.4 Contains() [1/2]

```
bool Contains (
    T value )
```

Returns true if the value already exists in the list.

6.212.3.5 Contains() [2/2]

```
bool Contains (
    T value,
    IEqualityComparer< T > comparer )
```

Returns true if the value already exists in the list.

6.212.3.6 Get()

```
T Get (
    int index )
```

Returns the value at supplied index.

6.212.3.7 GetEnumerator()

```
IEnumerator GetEnumerator ( )
```

Get the enumerator for the list.

6.212.3.8 IndexOf() [1/2]

```
int IndexOf (
    T value )
```

Returns the index with this value. Returns -1 if not found.

6.212.3.9 IndexOf() [2/2]

```
int IndexOf (
    T value,
    IEqualityComparer< T > equalityComparer )
```

Returns the index of the first occurrence of a value in the [NetworkLinkedList](#).

Parameters

<i>value</i>	The value to locate in the NetworkLinkedList . The value can be null for reference types.
<i>equalityComparer</i>	An equality comparer to compare values. Must not be null.

Returns

The zero-based index of the first occurrence of value within the entire [NetworkLinkedList](#), if found; otherwise, -1.

This method performs a linear search; therefore, this method is an O(n) operation, where n is Capacity.

6.212.3.10 Remap()

```
NetworkLinkedList<T> Remap (
    void * list )
```

Remaps the current [NetworkLinkedList](#) to a new memory location.

Parameters

<i>list</i>	The pointer to the new memory location.
-------------	---

Returns

A new instance of [NetworkLinkedList](#) with the same capacity and reader/writer, but mapped to the new memory location.

6.212.3.11 Remove() [1/2]

```
bool Remove (
    T value )
```

Removes the first found element with indicated value.

6.212.3.12 Remove() [2/2]

```
bool Remove (
    T value,
    IEqualityComparer< T > equalityComparer )
```

Removes the first found element with indicated value.

6.212.3.13 Set()

```
T Set (
    int index,
    T value )
```

Sets the value at supplied index.

6.212.4 Member Data Documentation

6.212.4.1 ELEMENT_WORDS

```
const int ELEMENT_WORDS = 2 [static]
```

Returns the number of words required to store a single element.

6.212.4.2 META_WORDS

```
const int META_WORDS = 3 [static]
```

Returns the number of words required to store the list metadata.

6.212.5 Property Documentation

6.212.5.1 Capacity

```
int Capacity [get]
```

Returns the max element count.

6.212.5.2 Count

```
int Count [get]
```

Returns the current element count.

6.212.5.3 this[int index]

T this[int index] [get], [set]

Element indexer.

6.213 NetworkLinkedList< T >.Enumerator Struct Reference

[Enumerator](#) for [NetworkLinkedList<T>](#).

Inherits [IEnumerator< T >](#).

Public Member Functions

- void [Dispose](#) ()
Releases all resources used by the [NetworkLinkedList<T>.Enumerator](#).
- bool [MoveNext](#) ()
Advances the enumerator to the next element of the [NetworkLinkedList<T>](#).
- void [Reset](#) ()
Resets the enumerator to its initial position, which is before the first element in the collection.

Public Attributes

- bool [_first](#)
- int [_head](#)
- [NetworkLinkedList< T >_list](#)

Properties

- T [Current](#) [get]
Gets the current element in the collection.
- object [IEnumerator](#). [Current](#) [get]

6.213.1 Detailed Description

[Enumerator](#) for [NetworkLinkedList<T>](#).

6.213.2 Member Function Documentation

6.213.2.1 Dispose()

```
void Dispose ( )
```

Releases all resources used by the NetworkLinkedList<T>.Enumerator.

6.213.2.2 MoveNext()

```
bool MoveNext ( )
```

Advances the enumerator to the next element of the [NetworkLinkedList<T>](#).

Returns

Returns true if the enumerator advanced to the next element.

6.213.2.3 Reset()

```
void Reset ( )
```

Resets the enumerator to its initial position, which is before the first element in the collection.

6.213.3 Property Documentation

6.213.3.1 Current

```
T Current [get]
```

Gets the current element in the collection.

Returns

The current element in the collection.

Exceptions

<i>InvalidOperationException</i>	Thrown when the enumerator is positioned before the first element or after the last element.
----------------------------------	--

6.214 NetworkLinkedListReadOnly< T > Struct Template Reference

Read-only version of NetworkLinkedList<T>.

Public Member Functions

- bool [Contains](#) (T value)
Returns true if the value already exists in the list.
- bool [Contains](#) (T value, IEqualityComparer< T > comparer)
Returns true if the value already exists in the list.
- int * **Entry** (int index)
- T [Get](#) (int index)
Returns the value at supplied index.
- int * **GetEntryByListIndex** (int listIndex)
- int [IndexOf](#) (T value)
Returns the index with this value. Returns -1 if not found.
- int [IndexOf](#) (T value, IEqualityComparer< T > equalityComparer)
Returns the index of the first occurrence of a value in the [FixedArray](#).
- T **Read** (int *entry)

Public Attributes

- int **_capacity**
- int * **_data**
- [IElementReaderWriter](#)< T > **_rw**
- int **_stride**

Static Public Attributes

- const int **COUNT** = 0
- const int [ELEMENT_WORDS](#) = 2
Returns the number of words required to store a single element.
- const int **HEAD** = 1
- const int **INVALID** = 0
- const int [META_WORDS](#) = 3
Returns the number of words required to store the list metadata.
- const int **NEXT** = 1
- const int **OFFSET** = 1
- const int **PREV** = 0
- const int **TAIL** = 2

Properties

- int [Capacity](#) [get]
Returns the max element count.
- int [Count](#) [get]
Returns the current element count.
- int **Head** [get]
- int **Tail** [get]
- T [this\[int index\]](#) [get]
Element indexer.

6.214.1 Detailed Description

Read-only version of NetworkLinkedList<T>.

Template Parameters

<i>T</i>	Custom struct type.
----------	---------------------

6.214.2 Member Function Documentation

6.214.2.1 Contains() [1/2]

```
bool Contains (
    T value )
```

Returns true if the value already exists in the list.

6.214.2.2 Contains() [2/2]

```
bool Contains (
    T value,
    IEqualityComparer< T > comparer )
```

Returns true if the value already exists in the list.

6.214.2.3 Get()

```
T Get (
    int index )
```

Returns the value at supplied index.

6.214.2.4 IndexOf() [1/2]

```
int IndexOf (
    T value )
```

Returns the index with this value. Returns -1 if not found.

6.214.2.5 IndexOf() [2/2]

```
int IndexOf (
    T value,
    IEqualityComparer< T > equalityComparer )
```

Returns the index of the first occurrence of a value in the [FixedArray](#).

Parameters

<i>value</i>	The value to locate in the FixedArray . The value can be null for reference types.
<i>equalityComparer</i>	An equality comparer to compare values. Must not be null.

Returns

The zero-based index of the first occurrence of value within the entire [FixedArray](#), if found; otherwise, -1.

This method performs a linear search; therefore, this method is an O(n) operation, where n is Capacity.

6.214.3 Member Data Documentation**6.214.3.1 ELEMENT_WORDS**

```
const int ELEMENT_WORDS = 2 [static]
```

Returns the number of words required to store a single element.

6.214.3.2 META_WORDS

```
const int META_WORDS = 3 [static]
```

Returns the number of words required to store the list metadata.

6.214.4 Property Documentation**6.214.4.1 Capacity**

```
int Capacity [get]
```

Returns the max element count.

6.214.4.2 Count

```
int Count [get]
```

Returns the current element count.

6.214.4.3 this[int index]

T this[int index] [get]

Element indexer.

6.215 NetworkLoadSceneParameters Struct Reference

Parameters for loading a scene

Inherits [IEquatable< NetworkLoadSceneParameters >](#).

Public Member Functions

- bool [Equals](#) ([NetworkLoadSceneParameters](#) other)
Compares two [NetworkLoadSceneParameters](#) for equality
- override bool [Equals](#) (object obj)
Compares two [NetworkLoadSceneParameters](#) for equality
- override int [GetHashCode](#) ()
Returns the hash code of the [NetworkLoadSceneParameters](#)
- override string [ToString](#) ()
Returns a string representation of the [NetworkLoadSceneParameters](#)

Static Public Member Functions

- static bool [operator!=](#) ([NetworkLoadSceneParameters](#) left, [NetworkLoadSceneParameters](#) right)
Compares two [NetworkLoadSceneParameters](#) for inequality
- static bool [operator==](#) ([NetworkLoadSceneParameters](#) left, [NetworkLoadSceneParameters](#) right)
Compares two [NetworkLoadSceneParameters](#) for equality

Public Attributes

- readonly [NetworkSceneLoadId](#) LoadId
The unique id of the scene load operation

Properties

- bool [IsActiveOnLoad](#) [get]
Signals if the scene should be active on load
- bool [IsLocalPhysics2D](#) [get]
Signals if the scene should have local 2D physics
- bool [IsLocalPhysics3D](#) [get]
Signals if the scene should have local 3D physics
- bool [IsSingleLoad](#) [get]
Signals if the scene should be single loaded
- LoadSceneMode [LoadSceneMode](#) [get]
The [LoadSceneMode](#) to use when loading the scene
- LoadSceneParameters [LoadSceneParameters](#) [get]
The [LoadSceneParameters](#) to use when loading the scene
- LocalPhysicsMode [LocalPhysicsMode](#) [get]
The [LocalPhysicsMode](#) to use when loading the scene

6.215.1 Detailed Description

Parameters for loading a scene

6.215.2 Member Function Documentation

6.215.2.1 Equals() [1/2]

```
bool Equals (  
    NetworkLoadSceneParameters other )
```

Compares two [NetworkLoadSceneParameters](#) for equality

Parameters

<i>other</i>	The other NetworkLoadSceneParameters
--------------	--

Returns

Returns true if the two [NetworkLoadSceneParameters](#) are equal

6.215.2.2 Equals() [2/2]

```
override bool Equals (  
    object obj )
```

Compares two [NetworkLoadSceneParameters](#) for equality

Parameters

<i>obj</i>	The other NetworkLoadSceneParameters
------------	--

Returns

Returns true if the two [NetworkLoadSceneParameters](#) are equal

6.215.2.3 GetHashCode()

```
override int GetHashCode ( )
```

Returns the hash code of the [NetworkLoadSceneParameters](#)

6.215.2.4 operator"!=(())

```
static bool operator!= (
    NetworkLoadSceneParameters left,
    NetworkLoadSceneParameters right ) [static]
```

Compares two [NetworkLoadSceneParameters](#) for inequality

Parameters

<i>left</i>	Left NetworkLoadSceneParameters
<i>right</i>	Right NetworkLoadSceneParameters

Returns

Returns true if the two [NetworkLoadSceneParameters](#) are not equal

6.215.2.5 operator==(())

```
static bool operator== (
    NetworkLoadSceneParameters left,
    NetworkLoadSceneParameters right ) [static]
```

Compares two [NetworkLoadSceneParameters](#) for equality

Parameters

<i>left</i>	Left NetworkLoadSceneParameters
<i>right</i>	Right NetworkLoadSceneParameters

Returns

Returns true if the two [NetworkLoadSceneParameters](#) are equal

6.215.2.6 ToString()

```
override string ToString ( )
```

Returns a string representation of the [NetworkLoadSceneParameters](#)

6.215.3 Member Data Documentation

6.215.3.1 LoadId

readonly [NetworkSceneLoadId](#) LoadId

The unique id of the scene load operation

6.215.4 Property Documentation

6.215.4.1 IsActiveOnLoad

bool IsActiveOnLoad [get]

Signals if the scene should be active on load

6.215.4.2 IsLocalPhysics2D

bool IsLocalPhysics2D [get]

Signals if the scene should have local 2D physics

6.215.4.3 IsLocalPhysics3D

bool IsLocalPhysics3D [get]

Signals if the scene should have local 3D physics

6.215.4.4 IsSingleLoad

bool IsSingleLoad [get]

Signals if the scene should be single loaded

6.215.4.5 LoadSceneMode

LoadSceneMode LoadSceneMode [get]

The [LoadSceneMode](#) to use when loading the scene

6.215.4.6 LoadSceneParameters

LoadSceneParameters LoadSceneParameters [get]

The [LoadSceneParameters](#) to use when loading the scene

6.215.4.7 LocalPhysicsMode

LocalPhysicsMode LocalPhysicsMode [get]

The [LocalPhysicsMode](#) to use when loading the scene

6.216 NetworkMecanimAnimator Class Reference

A component for synchronizing the Animator controller state from the State Authority to network proxies. Requires a Unity Animator component, and a [NetworkObject](#) component. NOTE: Animator Root Motion is not compatible with re-simulation and prediction.

Inherits [NetworkBehaviour](#), and [IAfterAllTicks](#).

Public Member Functions

- override void [Render](#) ()
Post simulation frame rendering callback. Runs after all simulations have finished. Use in place of Unity's Update when [Fusion](#) is handling Physics.
- void [SetTrigger](#) (int triggerHash, bool passThroughOnInputAuthority=false)
Queues a [SetTrigger\(\)](#) call for the associated Animator on the State Authority. Call this instead of [Animator.SetTrigger\(\)](#) for the State Authority to ensure that triggers are captured. On State Authority, this call will defer the [SetTrigger\(\)](#) pass-through to the Animator until [FixedUpdateNetwork\(\)](#) is called, where all queued triggers will be executed (this is to ensure tick agreement between server and clients).
- void [SetTrigger](#) (string trigger, bool passThroughOnInputAuthority=false)
Queues a [SetTrigger\(\)](#) call for the associated Animator on the State Authority. Call this instead of [Animator.SetTrigger\(\)](#) for the State Authority to ensure that triggers are captured. On State Authority, this call will defer the [SetTrigger\(\)](#) pass-through to the Animator until [FixedUpdateNetwork\(\)](#) is called, where all queued triggers will be executed (this is to ensure tick agreement between server and clients).
- override void [Spawned](#) ()
Post spawn callback.

Public Attributes

- Animator [Animator](#)
The Animator being synced. If unset, will attempt to find one on this GameObject.
- [RenderSource ApplyTiming](#) = [RenderSource.To](#)
The source of the State which is applied in Render.

Properties

- override? int [DynamicWordCount](#) [get]
Gets the dynamic word count for the [NetworkMecanimAnimator](#).

Additional Inherited Members

6.216.1 Detailed Description

A component for synchronizing the Animator controller state from the State Authority to network proxies. Requires a Unity Animator component, and a [NetworkObject](#) component. NOTE: Animator Root Motion is not compatible with re-simulation and prediction.

6.216.2 Member Function Documentation

6.216.2.1 SetTrigger() [1/2]

```
void SetTrigger (
    int triggerHash,
    bool passThroughOnInputAuthority = false )
```

Queues a [SetTrigger\(\)](#) call for the associated Animator on the State Authority. Call this instead of [Animator.SetTrigger\(\)](#) for the State Authority to ensure that triggers are captured. On State Authority, this call will defer the [SetTrigger\(\)](#) pass-through to the Animator until [FixedUpdateNetwork\(\)](#) is called, where all queued triggers will be executed (this is to ensure tick agreement between server and clients).

Parameters

<i>triggerHash</i>	Trigger hash to set
<i>passThroughOnInputAuthority</i>	Will call Animator.SetTrigger() immediately on the InputAuthority. If false, SetTrigger() will not be called on the Input Authority at all and Animator.SetTrigger() should be called explicitly as needed.

6.216.2.2 SetTrigger() [2/2]

```
void SetTrigger (
    string trigger,
    bool passThroughOnInputAuthority = false )
```

Queues a [SetTrigger\(\)](#) call for the associated Animator on the State Authority. Call this instead of [Animator.SetTrigger\(\)](#) for the State Authority to ensure that triggers are captured. On State Authority, this call will defer the [SetTrigger\(\)](#) pass-through to the Animator until [FixedUpdateNetwork\(\)](#) is called, where all queued triggers will be executed (this is to ensure tick agreement between server and clients).

Parameters

<i>trigger</i>	Trigger name to set
<i>passThroughOnInputAuthority</i>	Will call <code>Animator.SetTrigger()</code> immediately on the <code>InputAuthority</code> . If false, <code>SetTrigger()</code> will not be called on the <code>Input Authority</code> at all and <code>Animator.SetTrigger()</code> should be called explicitly as needed.

6.216.3 Member Data Documentation

6.216.3.1 Animator

```
Animator Animator
```

The Animator being synced. If unset, will attempt to find one on this `GameObject`.

6.216.3.2 ApplyTiming

```
RenderSource ApplyTiming = RenderSource.To
```

The source of the State which is applied in Render.

6.216.4 Property Documentation

6.216.4.1 DynamicWordCount

```
override? int DynamicWordCount [get]
```

Gets the dynamic word count for the [NetworkMecanimAnimator](#).

The dynamic word count, which is the maximum of the current total words and the runtime counts, if the application is playing.

Exceptions

<i>System.InvalidOperationException</i>	Thrown when this property is accessed outside of playing.
---	---

6.217 NetworkObject Class Reference

The primary [Fusion](#) component for networked `GameObject` entities. This stores the object's network identity and manages the object's state and input authority.

Inherits [Behaviour](#).

Public Member Functions

- void [AssignInputAuthority](#) ([PlayerRef](#) player)
Sets which [PlayerRef](#) has Input Authority for this Object.
- void [CopyStateFrom](#) ([NetworkObject](#) source)
Copies the entire State from another [NetworkObject](#)
- void [CopyStateFrom](#) ([NetworkObjectHeaderPtr](#) source)
Copies the entire State from another [NetworkObject](#) based on the [NetworkObjectHeaderPtr](#)
- int [GetLocalAuthorityMask](#) ()
Gets a bitmask of [AuthorityMasks](#) flags, representing the current local authority over this [NetworkObject](#).
- delegate [PriorityLevel](#) [PriorityLevelDelegate](#) ([NetworkObject](#) networkObject, [PlayerRef](#) player)
Delegate for determining the priority level of a network object for a specific player.
- void [ReleaseStateAuthority](#) ()
Release the state authority over this [NetworkObject](#) on shared mode.
- void [RemoveInputAuthority](#) ()
Removes input authority from whichever player has it for this object. Only valid when called on a Host or Server peer.
- delegate bool [ReplicateToDelegate](#) ([NetworkObject](#) networkObject, [PlayerRef](#) player)
Delegate for determining if a network object should be replicated to a specific player.
- void [RequestStateAuthority](#) ()
Request state authority over this [NetworkObject](#) on shared mode.
- void [SetPlayerAlwaysInterested](#) ([PlayerRef](#) player, bool alwaysInterested)
Add or remove specific player interest in this [NetworkObject](#). Only the [NetworkObject](#) State Authority can set interest.

Static Public Member Functions

- static int [GetWordCount](#) ([NetworkObject](#) obj)
Calculates the total word count for a given [NetworkObject](#).
- static void [NetworkUnwrap](#) ([NetworkRunner](#) runner, [NetworkId](#) wrapper, ref [NetworkObject](#) result)
Return the [NetworkObject](#) reference on result that matches the provided [NetworkId](#)
- static [NetworkId](#) [NetworkWrap](#) ([NetworkObject](#) obj)
Return the obj [NetworkId](#).
- static [NetworkId](#) [NetworkWrap](#) ([NetworkRunner](#) runner, [NetworkObject](#) obj)
Return the obj [NetworkId](#).
- static implicit operator [NetworkId](#) ([NetworkObject](#) obj)
Converts the Network Object to it's [NetworkId](#).

Public Attributes

- [NetworkObjectFlags](#) [Flags](#)
Flags used for network object prefabs and similar
- bool [ForceRemoteRenderTimeframe](#) = false
Force the [RenderTimeframe](#) used to [RenderTimeframe.Remote](#)
- bool [IsResume](#)
Signal that this [NetworkObject](#) comes from a Resume Spawn
- [NetworkObject\[\]](#) [NestedObjects](#)
Array of initial child nested [NetworkObject](#) entities, that are children of this Object.
- [NetworkBehaviour\[\]](#) [NetworkedBehaviours](#)
Array of all [NetworkBehaviours](#) associated with this network entity.
- [NetworkObjectTypeId](#) [NetworkTypeId](#)
The type ID for this prefab or scene object, set when adding to the prefab table and registering scene objects, respectively. All spawned instances of this object will retain this value. Use [NetworkId](#) for the unique ID of network entries.
- [PriorityLevelDelegate](#) [PriorityCallback](#)
Delegate callback used to override priority value for a specific object-player pair
- [ReplicateToDelegate](#) [ReplicateTo](#)
Delegate callback used to override if an object should be replicate to a client or not
- uint [SortKey](#)
Used for whenever objects need to be sorted in a deterministic order, like when registering scene objects.

Protected Member Functions

- virtual void [Awake](#) ()
Awake is called when the script instance is being loaded.
- virtual void [OnDestroy](#) ()
OnDestroy is called when the script instance is being destroyed.

Properties

- bool [HasInputAuthority](#) [get]
Returns if [Simulation.LocalPlayer](#) is the designated Input Source for this network entity.
- bool [HasStateAuthority](#) [get]
Returns if [Simulation.LocalPlayer](#) is the designated State Source for this network entity.
- [NetworkId?](#) [Id](#) [get]
The unique identifier for this network entity.
- [PlayerRef](#) [InputAuthority](#) [get]
Returns the [PlayerRef](#) that has Input Authority over this network entity. PlayerRefs are assigned in order from 0 to [MaxPlayers-1](#) and are re-used as players join and leave. The only caveat is that the server player (if one exists), always gets the last index no matter how many clients are connected.
- bool [IsInSimulation](#) [get]
If this object is inserted into the simulation
- bool [IsNested](#) [get]
Returns true if this object is nested in a prefab instance.
- bool [IsProxy](#) [get]
Returns if [Simulation.LocalPlayer](#) is neither the Input nor State Source for this network entity.
- bool [IsSpawnable](#) [get, set]

Toggles if this [NetworkObject](#) is included in the [NetworkProjectConfig.PrefabTable](#), which will include the prefab in builds as a [Spawnable object](#).

- [bool IsValid](#) [get]

Returns if this network entity is associated with its [NetworkRunner](#), and that runner is not null.
- [Tick LastReceiveTick](#) [get]

Last tick this object received an update.
- [string Name](#) [get]

The ID + Unity [GameObject](#) name for this entity.
- [NetworkObject NestingRoot](#) [get]

Returns root of this prefab instance, if this object is nested.
- [RenderSource RenderSource](#) [get, set]

Returns the [Fusion.RenderSource](#) for this [Fusion.NetworkBehaviour](#) instance, indicating how snapshot data will be used to render it.
- [float RenderTime](#) [get]

Returns the current interpolation time for this object
- [RenderTimeframe?? RenderTimeframe](#) [get]

Returns the [Fusion.RenderTimeframe](#) for this [Fusion.NetworkBehaviour](#) instance, indicating what snapshot data will be used to render it.
- [NetworkRunner Runner](#) [get]

The [NetworkRunner](#) this entity is associated with.
- [PlayerRef StateAuthority](#) [get]

Returns the [PlayerRef](#) that has State Authority over this network entity. [PlayerRefs](#) are assigned in order from 0 to [MaxPlayers-1](#) and are re-used as players join and leave. The only caveat is that the server player (if one exists), always gets the last index no matter how many clients are connected.

6.217.1 Detailed Description

The primary [Fusion](#) component for networked [GameObject](#) entities. This stores the object's network identity and manages the object's state and input authority.

6.217.2 Member Function Documentation

6.217.2.1 AssignInputAuthority()

```
void AssignInputAuthority (
    PlayerRef player )
```

Sets which [PlayerRef](#) has Input Authority for this Object.

6.217.2.2 Awake()

```
virtual void Awake ( ) [protected], [virtual]
```

Awake is called when the script instance is being loaded.

6.217.2.3 CopyStateFrom() [1/2]

```
void CopyStateFrom (
    NetworkObject source )
```

Copies the entire State from another [NetworkObject](#)

Parameters

<i>source</i>	NetworkObject to copy the State from
---------------	--

6.217.2.4 CopyStateFrom() [2/2]

```
void CopyStateFrom (
    NetworkObjectHeaderPtr source )
```

Copies the entire State from another [NetworkObject](#) based on the [NetworkObjectHeaderPtr](#)

Parameters

<i>source</i>	NetworkObjectHeaderPtr to copy the state from
---------------	---

6.217.2.5 GetLocalAuthorityMask()

```
int GetLocalAuthorityMask ( )
```

Gets a bitmask of [AuthorityMasks](#) flags, representing the current local authority over this [NetworkObject](#).

6.217.2.6 GetWordCount()

```
static int GetWordCount (
    NetworkObject obj ) [static]
```

Calculates the total word count for a given [NetworkObject](#).

Parameters

<i>obj</i>	The NetworkObject for which the word count is to be calculated.
------------	---

Returns

The total word count of the [NetworkObject](#). Returns 0 if the [NetworkObject](#) is not alive.

Exceptions

<i>System.Exception</i>	Thrown when a NetworkBehaviour reference is missing in the NetworkBehaviour array of the NetworkObject .
-------------------------	--

6.217.2.7 NetworkUnwrap()

```
static void NetworkUnwrap (
    NetworkRunner runner,
    NetworkId wrapper,
    ref NetworkObject result ) [static]
```

Return the [NetworkObject](#) reference on *result* that matches the provided [NetworkId](#)

Parameters

<i>runner</i>	The NetworkRunner that will be used to try to find a NetworkObject with ID equals to <i>wrapper</i>
<i>wrapper</i>	The NetworkId to be searched
<i>result</i>	The found NetworkObject . null if the provided NetworkId is not valid

6.217.2.8 NetworkWrap() [1/2]

```
static NetworkId NetworkWrap (
    NetworkObject obj ) [static]
```

Return the *obj* [NetworkId](#).

Parameters

<i>obj</i>	The NetworkObject to get the ID from
------------	--

Returns

The [NetworkId](#) of the object. Default if the object is not alive (null or destroyed)

6.217.2.9 NetworkWrap() [2/2]

```
static NetworkId NetworkWrap (
    NetworkRunner runner,
    NetworkObject obj ) [static]
```

Return the *obj* [NetworkId](#).

Parameters

<i>runner</i>	The NetworkRunner that <i>obj</i> is assigned to
<i>obj</i>	The NetworkObject to get the ID from

Returns

The [NetworkId](#) of the object. Default if the object is not alive (null or destroyed)

6.217.2.10 OnDestroy()

```
virtual void OnDestroy ( ) [protected], [virtual]
```

OnDestroy is called when the script instance is being destroyed.

6.217.2.11 operator NetworkId()

```
static implicit operator NetworkId (
    NetworkObject obj ) [static]
```

Converts the Network Object to its [NetworkId](#).

Parameters

<i>obj</i>	The object to convert
------------	-----------------------

Returns

The [NetworkId](#) of the object. Default if the object is not alive (null or destroyed)

6.217.2.12 PriorityLevelDelegate()

```
delegate PriorityLevel PriorityLevelDelegate (
    NetworkObject networkObject,
    PlayerRef player )
```

Delegate for determining the priority level of a network object for a specific player.

Parameters

<i>networkObject</i>	The network object in question.
<i>player</i>	The player for whom the priority level is being determined.

Returns

The priority level of the network object for the player.

6.217.2.13 ReleaseStateAuthority()

```
void ReleaseStateAuthority ( )
```

Release the state authority over this [NetworkObject](#) on shared mode.

6.217.2.14 RemoveInputAuthority()

```
void RemoveInputAuthority ( )
```

Removes input authority from whichever player has it for this object. Only valid when called on a Host or Server peer.

6.217.2.15 ReplicateToDelegate()

```
delegate bool ReplicateToDelegate (
    NetworkObject networkObject,
    PlayerRef player )
```

Delegate for determining if a network object should be replicated to a specific player.

Parameters

<i>networkObject</i>	The network object in question.
<i>player</i>	The player to potentially replicate to.

Returns

True if the object should be replicated to the player, false otherwise.

6.217.2.16 RequestStateAuthority()

```
void RequestStateAuthority ( )
```

Request state authority over this [NetworkObject](#) on shared mode.

6.217.2.17 SetPlayerAlwaysInterested()

```
void SetPlayerAlwaysInterested (
    PlayerRef player,
    bool alwaysInterested )
```

Add or remove specific player interest in this [NetworkObject](#). Only the [NetworkObject](#) State Authority can set interest.

Parameters

<i>player</i>	The player to set interest for
<i>alwaysInterested</i>	If the player should always be interested in this object

6.217.3 Member Data Documentation

6.217.3.1 Flags

`NetworkObjectFlags` Flags

Flags used for network object prefabs and similar

6.217.3.2 ForceRemoteRenderTimeframe

`bool ForceRemoteRenderTimeframe = false`

Force the `RenderTimeframe` used to `RenderTimeframe.Remote`

6.217.3.3 IsResume

`bool IsResume`

Signal that this `NetworkObject` comes from a Resume Spawn

6.217.3.4 NestedObjects

`NetworkObject [] NestedObjects`

Array of initial child nested `NetworkObject` entities, that are children of this Object.

6.217.3.5 NetworkedBehaviours

`NetworkBehaviour [] NetworkedBehaviours`

Array of all `NetworkBehaviour`s associated with this network entity.

6.217.3.6 NetworkTypeId

`NetworkObjectId` `NetworkTypeId`

The type ID for this prefab or scene object, set when adding to the prefab table and registering scene objects, respectively. All spawned instances of this object will retain this value. Use [NetworkId](#) for the unique ID of network entries.

6.217.3.7 PriorityCallback

`PriorityLevelDelegate` `PriorityCallback`

Delegate callback used to override priority value for a specific object-player pair

6.217.3.8 ReplicateTo

`ReplicateToDelegate` `ReplicateTo`

Delegate callback used to override if an object should be replicate to a client or not

6.217.3.9 SortKey

`uint` `SortKey`

Used for whenever objects need to be sorted in a deterministic order, like when registering scene objects.

6.217.4 Property Documentation

6.217.4.1 HasInputAuthority

`bool` `HasInputAuthority` `[get]`

Returns if [Simulation.LocalPlayer](#) is the designated Input Source for this network entity.

6.217.4.2 HasStateAuthority

```
bool HasStateAuthority [get]
```

Returns if [Simulation.LocalPlayer](#) is the designated State Source for this network entity.

6.217.4.3 Id

```
NetworkId? Id [get]
```

The unique identifier for this network entity.

6.217.4.4 InputAuthority

```
PlayerRef InputAuthority [get]
```

Returns the [PlayerRef](#) that has Input Authority over this network entity. PlayerRefs are assigned in order from 0 to MaxPlayers-1 and are re-used as players join and leave. The only caveat is that the server player (if one exists), always gets the last index no matter how many clients are connected.

6.217.4.5 IsInSimulation

```
bool IsInSimulation [get]
```

If this object is inserted into the simulation

6.217.4.6 IsNested

```
bool IsNested [get]
```

Returns true if this object is nested in a prefab instance.

6.217.4.7 IsProxy

```
bool IsProxy [get]
```

Returns if [Simulation.LocalPlayer](#) is neither the Input nor State Source for this network entity.

6.217.4.8 IsSpawnable

```
bool IsSpawnable [get], [set]
```

Toggles if this [NetworkObject](#) is included in the [NetworkProjectConfig.PrefabTable](#), which will include the prefab in builds as a Spawnable object.

6.217.4.9 IsValid

```
bool IsValid [get]
```

Returns if this network entity is associated with its [NetworkRunner](#), and that runner is not null.

6.217.4.10 LastReceiveTick

```
Tick LastReceiveTick [get]
```

Last tick this object received an update.

6.217.4.11 Name

```
string Name [get]
```

The ID + Unity GameObject name for this entity.

6.217.4.12 NestingRoot

```
NetworkObject NestingRoot [get]
```

Returns root of this prefab instance, if this object is nested.

6.217.4.13 RenderSource

```
RenderSource RenderSource [get], [set]
```

Returns the [Fusion.RenderSource](#) for this [Fusion.NetworkBehaviour](#) instance, indicating how snapshot data will be used to render it.

6.217.4.14 RenderTime

```
float RenderTime [get]
```

Returns the current interpolation time for this object

6.217.4.15 RenderTimeframe

```
RenderTimeframe?? RenderTimeframe [get]
```

Returns the [Fusion.RenderTimeframe](#) for this [Fusion.NetworkBehaviour](#) instance, indicating what snapshot data will be used to render it.

6.217.4.16 Runner

```
NetworkRunner Runner [get]
```

The [NetworkRunner](#) this entity is associated with.

6.217.4.17 StateAuthority

```
PlayerRef StateAuthority [get]
```

Returns the [PlayerRef](#) that has State Authority over this network entity. PlayerRefs are assigned in order from 0 to MaxPlayers-1 and are re-used as players join and leave. The only caveat is that the server player (if one exists), always gets the last index no matter how many clients are connected.

6.218 NetworkObjectFlagsExtensions Class Reference

Extension methods for the NetworkObjectFlags enum.

Static Public Member Functions

- static int [GetVersion](#) (this [NetworkObjectFlags](#) flags)
Returns the version of the flags.
- static bool [IsIgnored](#) (this [NetworkObjectFlags](#) flags)
Check if the flags are ignored.
- static bool [IsVersionCurrent](#) (this [NetworkObjectFlags](#) flags)
Check if the flags are the current version.
- static [NetworkObjectFlags SetCurrentVersion](#) (this [NetworkObjectFlags](#) flags)
Sets the flags to the current version.
- static [NetworkObjectFlags SetIgnored](#) (this [NetworkObjectFlags](#) flags, bool value)
Sets the ignored flag on the flags.

6.218.1 Detailed Description

Extension methods for the NetworkObjectFlags enum.

6.218.2 Member Function Documentation

6.218.2.1 GetVersion()

```
static int GetVersion (  
    this NetworkObjectFlags flags ) [static]
```

Returns the version of the flags.

Parameters

<i>flags</i>	The flags to get the version of.
--------------	----------------------------------

Returns

The version of the flags.

6.218.2.2 IsIgnored()

```
static bool IsIgnored (  
    this NetworkObjectFlags flags ) [static]
```

Check if the flags are ignored.

Parameters

<i>flags</i>	The flags to check.
--------------	---------------------

Returns

True if the flags are ignored, false otherwise.

6.218.2.3 IsVersionCurrent()

```
static bool IsVersionCurrent (  
    this NetworkObjectFlags flags ) [static]
```

Check if the flags are the current version.

Parameters

<i>flags</i>	The flags to check.
--------------	---------------------

Returns

True if the flags are the current version, false otherwise.

6.218.2.4 SetCurrentVersion()

```
static NetworkObjectFlags SetCurrentVersion (
    this NetworkObjectFlags flags ) [static]
```

Sets the flags to the current version.

Parameters

<i>flags</i>	The flags to set.
--------------	-------------------

Returns

The flags with the version set to the current version.

6.218.2.5 SetIgnored()

```
static NetworkObjectFlags SetIgnored (
    this NetworkObjectFlags flags,
    bool value ) [static]
```

Sets the ignored flag on the flags.

Parameters

<i>flags</i>	Flags to set the ignored flag on.
<i>value</i>	Ignored flag value.

Returns

The flags with the ignored flag set to the given value.

6.219 NetworkObjectGuid Struct Reference

[NetworkObjectGuid](#)

Inherits [INetworkStruct](#), [IEquatable< NetworkObjectGuid >](#), and [IComparable< NetworkObjectGuid >](#).

Classes

- class [EqualityComparer](#)
EqualityComparer for NetworkObjectGuid

Public Member Functions

- int [CompareTo](#) ([NetworkObjectGuid](#) other)
Compare the [NetworkObjectGuid](#) to another [NetworkObjectGuid](#)
- bool [Equals](#) ([NetworkObjectGuid](#) other)
Check if the [NetworkObjectGuid](#) is equal to another [NetworkObjectGuid](#)
- override bool [Equals](#) (object obj)
Check if the [NetworkObjectGuid](#) is equal to another object
- override int [GetHashCode](#) ()
Get the hashcode for a [NetworkObjectGuid](#)
- [NetworkObjectGuid](#) (byte *guid)
*Create a [NetworkObjectGuid](#) from a byte**
- [NetworkObjectGuid](#) (byte[] guid)
Create a [NetworkObjectGuid](#) from a byte array
- [NetworkObjectGuid](#) (long data0, long data1)
Create a new [NetworkObjectGuid](#)
- [NetworkObjectGuid](#) (string guid)
Create a new [NetworkObjectGuid](#)
- override string [ToString](#) ()
Returns a string representation of the [NetworkObjectGuid](#).
- string [ToString](#) (string format)
Returns a string representation of the [NetworkObjectGuid](#).
- string [ToUnityGuidString](#) ()
Returns a string representation of the [NetworkObjectGuid](#).

Static Public Member Functions

- static implicit operator [Guid](#) ([NetworkObjectGuid](#) guid)
Implicit conversion from [NetworkObjectGuid](#) to [Guid](#)
- static implicit operator [NetworkObjectGuid](#) ([Guid](#) guid)
Implicit conversion from [Guid](#) to [NetworkObjectGuid](#)
- static operator [NetworkPrefabRef](#) ([NetworkObjectGuid](#) t)
Explicit conversion from [NetworkObjectGuid](#) to [NetworkPrefabRef](#)
- static bool operator != ([NetworkObjectGuid](#) a, [NetworkObjectGuid](#) b)
Compare two [NetworkObjectGuid](#)
- static bool operator == ([NetworkObjectGuid](#) a, [NetworkObjectGuid](#) b)
Compare two [NetworkObjectGuid](#)
- static [NetworkObjectGuid](#) Parse (string str)
Parse a [NetworkObjectGuid](#) from a string.
- static bool TryParse (string str, out [NetworkObjectGuid](#) guid)
Try to parse a string into a [NetworkObjectGuid](#)

Public Attributes

- fixed long [RawGuidValue](#) [2]
The Raw Guid Value of the [NetworkObjectGuid](#)

Static Public Attributes

- const int [ALIGNMENT](#) = 4
The alignment of the [NetworkObjectGuid](#)
- const int [SIZE](#) = 16
The Size of the [NetworkObjectGuid](#) in bytes

Properties

- static [NetworkObjectGuid Empty](#) [get]
The default value of a [NetworkObjectGuid](#)
- bool [IsValid](#) [get]
Signal if the [NetworkObjectGuid](#) is valid.

6.219.1 Detailed Description

[NetworkObjectGuid](#)

6.219.2 Constructor & Destructor Documentation

6.219.2.1 NetworkObjectGuid() [1/4]

```
NetworkObjectGuid (
    string guid )
```

Create a new [NetworkObjectGuid](#)

Parameters

<i>guid</i>	The guid to use
-------------	-----------------

6.219.2.2 NetworkObjectGuid() [2/4]

```
NetworkObjectGuid (
    long data0,
    long data1 )
```


Create a new [NetworkObjectGuid](#)

Parameters

<i>data0</i>	Data0 of the Guid
<i>data1</i>	Data1 of the Guid

6.219.2.3 NetworkObjectGuid() [3/4]

```
NetworkObjectGuid (
    byte[] guid )
```

Create a [NetworkObjectGuid](#) from a byte array

Parameters

<i>guid</i>	The byte array to create the NetworkObjectGuid from
-------------	---

6.219.2.4 NetworkObjectGuid() [4/4]

```
NetworkObjectGuid (
    byte * guid )
```

Create a [NetworkObjectGuid](#) from a byte*

Parameters

<i>guid</i>	The byte* to create the NetworkObjectGuid from
-------------	--

6.219.3 Member Function Documentation**6.219.3.1 CompareTo()**

```
int CompareTo (
    NetworkObjectGuid other )
```

Compare the [NetworkObjectGuid](#) to another [NetworkObjectGuid](#)

Parameters

<i>other</i>	The other NetworkObjectGuid to compare against
--------------	--

Returns

0 if the [NetworkObjectGuid](#) are equal, -1 if this [NetworkObjectGuid](#) is less than the other, 1 if this [NetworkObjectGuid](#) is greater than the other

6.219.3.2 Equals() [1/2]

```
bool Equals (  
    NetworkObjectGuid other )
```

Check if the [NetworkObjectGuid](#) is equal to another [NetworkObjectGuid](#)

Parameters

<i>other</i>	The other NetworkObjectGuid to check against
--------------	--

Returns

True if the [NetworkObjectGuid](#)s are equal, false otherwise

6.219.3.3 Equals() [2/2]

```
override bool Equals (  
    object obj )
```

Check if the [NetworkObjectGuid](#) is equal to another object

Parameters

<i>obj</i>	The other object to check against
------------	-----------------------------------

Returns

True if the objects are equal, false otherwise

6.219.3.4 GetHashCode()

```
override int GetHashCode ( )
```

Get the hashcode for a [NetworkObjectGuid](#)

6.219.3.5 operator Guid()

```
static implicit operator Guid (  
    NetworkObjectGuid guid ) [static]
```

Implicit conversion from [NetworkObjectGuid](#) to Guid

Parameters

<i>guid</i>	NetworkObjectGuid to convert from
-------------	---

Returns

Guid

6.219.3.6 operator NetworkObjectGuid()

```
static implicit operator NetworkObjectGuid (  
    Guid guid ) [static]
```

Implicit conversion from Guid to [NetworkObjectGuid](#)

Parameters

<i>guid</i>	Guid to convert from
-------------	----------------------

Returns

[NetworkObjectGuid](#)

6.219.3.7 operator NetworkPrefabRef()

```
static operator NetworkPrefabRef (  
    NetworkObjectGuid t ) [explicit], [static]
```

Explicit conversion from [NetworkObjectGuid](#) to [NetworkPrefabRef](#)

Parameters

<i>t</i>	NetworkObjectGuid to convert from
----------	---

Returns

[NetworkPrefabRef](#)

6.219.3.8 operator"!="()

```
static bool operator!= (
    NetworkObjectGuid a,
    NetworkObjectGuid b ) [static]
```

Compare two [NetworkObjectGuid](#)

Returns

True if the [NetworkObjectGuid](#) are not equal, false otherwise

6.219.3.9 operator=="()

```
static bool operator== (
    NetworkObjectGuid a,
    NetworkObjectGuid b ) [static]
```

Compare two [NetworkObjectGuid](#)

Returns

True if the [NetworkObjectGuid](#) are equal, false otherwise

6.219.3.10 Parse()

```
static NetworkObjectGuid Parse (
    string str ) [static]
```

Parse a [NetworkObjectGuid](#) from a string.

Parameters

<i>str</i>	The string to parse.
------------	----------------------

Returns

The parsed [NetworkObjectGuid](#).

6.219.3.11 ToString() [1/2]

```
override string ToString ( )
```

Returns a string representation of the [NetworkObjectGuid](#).

6.219.3.12 ToString() [2/2]

```
string ToString (
    string format )
```

Returns a string representation of the [NetworkObjectGuid](#).

6.219.3.13 ToUnityGuidString()

```
string ToUnityGuidString ( )
```

Returns a string representation of the [NetworkObjectGuid](#).

6.219.3.14 TryParse()

```
static bool TryParse (
    string str,
    out NetworkObjectGuid guid ) [static]
```

Try to parse a string into a [NetworkObjectGuid](#)

Parameters

<i>str</i>	String to parse
<i>guid</i>	Parsed NetworkObjectGuid

Returns

True if the string was parsed successfully, false otherwise

6.219.4 Member Data Documentation

6.219.4.1 ALIGNMENT

```
const int ALIGNMENT = 4 [static]
```

The alignment of the [NetworkObjectGuid](#)

6.219.4.2 RawGuidValue

```
fixed long RawGuidValue[2]
```

The Raw Guid Value of the [NetworkObjectGuid](#)

6.219.4.3 SIZE

```
const int SIZE = 16 [static]
```

The Size of the [NetworkObjectGuid](#) in bytes

6.219.5 Property Documentation

6.219.5.1 Empty

```
NetworkObjectGuid Empty [static], [get]
```

The default value of a [NetworkObjectGuid](#)

6.219.5.2 IsValid

```
bool IsValid [get]
```

Signal if the [NetworkObjectGuid](#) is valid.

6.220 NetworkObjectGuid.EqualityComparer Class Reference

[EqualityComparer](#) for [NetworkObjectGuid](#)

Inherits [IEqualityComparer](#)< [NetworkObjectGuid](#) >.

Public Member Functions

- bool [Equals](#) ([NetworkObjectGuid](#) x, [NetworkObjectGuid](#) y)
Check if two [NetworkObjectGuid](#) are equals
- int [GetHashCode](#) ([NetworkObjectGuid](#) obj)
Get the hashcode for a [NetworkObjectGuid](#)

6.220.1 Detailed Description

[EqualityComparer](#) for [NetworkObjectGuid](#)

6.220.2 Member Function Documentation

6.220.2.1 Equals()

```
bool Equals (
    NetworkObjectGuid x,
    NetworkObjectGuid y )
```

Check if two [NetworkObjectGuid](#) are equals

6.220.2.2 GetHashCode()

```
int GetHashCode (
    NetworkObjectGuid obj )
```

Get the hashcode for a [NetworkObjectGuid](#)

6.221 NetworkObjectHeader Struct Reference

Network object header information for a [NetworkObject](#).

Inherits [INetworkStruct](#), and [IEquatable< NetworkObjectHeader >](#).

Public Member Functions

- bool [Equals](#) ([NetworkObjectHeader](#) other)
Checks if the current instance of [NetworkObjectHeader](#) is equal to another instance of the same type.
- override bool [Equals](#) (object obj)
Checks if the current instance of [NetworkObjectHeader](#) is equal to another object.
- override int [GetHashCode](#) ()
Generates a hash code for the current instance of [NetworkObjectHeader](#).
- override string [ToString](#) ()
The string representation of the [NetworkObjectHeader](#).

Static Public Member Functions

- static int * [GetBehaviourChangedTickArray](#) ([NetworkObjectHeader](#) *header)
Returns a pointer to the array of behaviour change ticks in a [NetworkObjectHeader](#).
- static int * [GetDataPointer](#) ([NetworkObjectHeader](#) *header)
Returns a pointer to the data of a [NetworkObjectHeader](#).
- static int [GetDataWordCount](#) ([NetworkObjectHeader](#) *header)
Returns the count of data words in a [NetworkObjectHeader](#).
- static [NetworkTRSPData](#) * [GetMainNetworkTRSPData](#) ([NetworkObjectHeader](#) *header)
Returns a pointer to the main network TRSP data of a [NetworkObjectHeader](#), if it exists.
- static bool [HasMainNetworkTRSP](#) ([NetworkObjectHeader](#) *header)
Checks if a [NetworkObjectHeader](#) has a main network TRSP.
- static bool [operator!=](#) ([NetworkObjectHeader](#) left, [NetworkObjectHeader](#) right)
Determines if two instances of [NetworkObjectHeader](#) are not equal.
- static bool [operator==](#) ([NetworkObjectHeader](#) left, [NetworkObjectHeader](#) right)
Determines if two instances of [NetworkObjectHeader](#) are equal.

Public Attributes

- fixed int [_reserved](#) [10]
Reserved space for future use.
- short [BehaviourCount](#)
The number of behaviours in the network object.
- [NetworkObjectHeaderFlags](#) [Flags](#)
The flags indicating various states or properties of the network object.
- [NetworkId](#) [Id](#)
The unique identifier of the network object.
- [PlayerRef](#) [InputAuthority](#)
The player reference who has input authority over the network object.
- [NetworkObjectNestingKey](#) [NestingKey](#)
The nesting key of the network object.
- [NetworkId](#) [NestingRoot](#)
The unique identifier of the root network object in the nesting hierarchy.
- [PlayerRef](#) [StateAuthority](#)
The player reference who has state authority over the network object.
- [NetworkObjectTypeId](#) [Type](#)
The type identifier of the network object.
- short [WordCount](#)
The size of the network object's data in words.

Static Public Attributes

- const int [PLAYER_DATA_WORD](#) = 36 / [Allocator.REPLICATE_WORD_SIZE](#)
The word index of the player data in the [NetworkObjectHeader](#).
- const int [SIZE](#) = 80
The size of the [NetworkObjectHeader](#) in bytes.
- const int [WORDS](#) = [SIZE](#) / [Allocator.REPLICATE_WORD_SIZE](#)
The size of the [NetworkObjectHeader](#) in words.

Properties

- int [ByteCount](#) [get]
The size of the network object's data in bytes.

6.221.1 Detailed Description

Network object header information for a [NetworkObject](#).

6.221.2 Member Function Documentation

6.221.2.1 Equals() [1/2]

```
bool Equals (  
    NetworkObjectHeader other )
```

Checks if the current instance of [NetworkObjectHeader](#) is equal to another instance of the same type.

6.221.2.2 Equals() [2/2]

```
override bool Equals (  
    object obj )
```

Checks if the current instance of [NetworkObjectHeader](#) is equal to another object.

6.221.2.3 GetBehaviourChangedTickArray()

```
static int* GetBehaviourChangedTickArray (  
    NetworkObjectHeader * header ) [static]
```

Returns a pointer to the array of behaviour change ticks in a [NetworkObjectHeader](#).

Parameters

<i>header</i>	Pointer to the NetworkObjectHeader .
---------------	--

Returns

Pointer to the array of behaviour change ticks in the [NetworkObjectHeader](#).

6.221.2.4 GetDataPointer()

```
static int* GetDataPointer (
    NetworkObjectHeader * header ) [static]
```

Returns a pointer to the data of a [NetworkObjectHeader](#).

Parameters

<i>header</i>	Pointer to the NetworkObjectHeader .
---------------	--

Returns

Pointer to the data of the [NetworkObjectHeader](#).

6.221.2.5 GetDataWordCount()

```
static int GetDataWordCount (
    NetworkObjectHeader * header ) [static]
```

Returns the count of data words in a [NetworkObjectHeader](#).

Parameters

<i>header</i>	Pointer to the NetworkObjectHeader .
---------------	--

Returns

The count of data words in the [NetworkObjectHeader](#).

6.221.2.6 GetHashCode()

```
override int GetHashCode ( )
```

Generates a hash code for the current instance of [NetworkObjectHeader](#).

6.221.2.7 GetMainNetworkTRSPData()

```
static NetworkTRSPData* GetMainNetworkTRSPData (
    NetworkObjectHeader * header ) [static]
```

Returns a pointer to the main network TRSP data of a [NetworkObjectHeader](#), if it exists.

Parameters

<i>header</i>	Pointer to the NetworkObjectHeader .
---------------	--

Returns

Pointer to the main network TRSP data of the [NetworkObjectHeader](#) if it exists, null otherwise.

6.221.2.8 HasMainNetworkTRSP()

```
static bool HasMainNetworkTRSP (  
    NetworkObjectHeader * header ) [static]
```

Checks if a [NetworkObjectHeader](#) has a main network TRSP.

Parameters

<i>header</i>	Pointer to the NetworkObjectHeader .
---------------	--

Returns

True if the [NetworkObjectHeader](#) has a main network TRSP, false otherwise.

6.221.2.9 operator"!="()

```
static bool operator!= (  
    NetworkObjectHeader left,  
    NetworkObjectHeader right ) [static]
```

Determines if two instances of [NetworkObjectHeader](#) are not equal.

Returns

True if the instances are not equal; otherwise, false.

6.221.2.10 operator=="()

```
static bool operator== (  
    NetworkObjectHeader left,  
    NetworkObjectHeader right ) [static]
```

Determines if two instances of [NetworkObjectHeader](#) are equal.

Returns

True if the instances are equal; otherwise, false.

6.221.2.11 ToString()

```
override string ToString ( )
```

The string representation of the [NetworkObjectHeader](#).

6.221.3 Member Data Documentation

6.221.3.1 _reserved

```
fixed int _reserved[10]
```

Reserved space for future use.

6.221.3.2 BehaviourCount

```
short BehaviourCount
```

The number of behaviours in the network object.

6.221.3.3 Flags

```
NetworkObjectHeaderFlags Flags
```

The flags indicating various states or properties of the network object.

6.221.3.4 Id

```
NetworkId Id
```

The unique identifier of the network object.

6.221.3.5 InputAuthority

```
PlayerRef InputAuthority
```

The player reference who has input authority over the network object.

6.221.3.6 NestingKey

[NetworkObjectNestingKey](#) `NestingKey`

The nesting key of the network object.

6.221.3.7 NestingRoot

[NetworkId](#) `NestingRoot`

The unique identifier of the root network object in the nesting hierarchy.

6.221.3.8 PLAYER_DATA_WORD

```
const int PLAYER_DATA_WORD = 36 / Allocator.REPLICATE\_WORD\_SIZE [static]
```

The word index of the player data in the [NetworkObjectHeader](#).

6.221.3.9 SIZE

```
const int SIZE = 80 [static]
```

The size of the [NetworkObjectHeader](#) in bytes.

6.221.3.10 StateAuthority

[PlayerRef](#) `StateAuthority`

The player reference who has state authority over the network object.

6.221.3.11 Type

[NetworkObjectTypeId](#) `Type`

The type identifier of the network object.

6.221.3.12 WordCount

```
short WordCount
```

The size of the network object's data in words.

6.221.3.13 WORDS

```
const int WORDS = SIZE / Allocator.REPLICATE_WORD_SIZE [static]
```

The size of the [NetworkObjectHeader](#) in words.

6.221.4 Property Documentation

6.221.4.1 ByteCount

```
int ByteCount [get]
```

The size of the network object's data in bytes.

6.222 NetworkObjectHeaderPtr Struct Reference

Represents a pointer to a [NetworkObjectHeader](#). This struct is unsafe because it uses pointers.

Static Public Member Functions

- static implicit [operator NetworkObjectHeaderPtr](#) ([NetworkObjectHeader](#) *ptr)

Public Attributes

- [NetworkObjectHeader](#) * Ptr
Pointer to a [NetworkObjectHeader](#).

Properties

- [NetworkId](#) Id [get]
Gets the Id of the [NetworkObjectHeader](#) this struct points to.
- [NetworkObjectType](#) Id Type [get]
Gets the Type of the [NetworkObjectHeader](#) this struct points to.

6.222.1 Detailed Description

Represents a pointer to a [NetworkObjectHeader](#). This struct is unsafe because it uses pointers.

6.222.2 Member Function Documentation

6.222.2.1 operator NetworkObjectHeaderPtr()

```
static implicit operator NetworkObjectHeaderPtr (  
    NetworkObjectHeader * ptr ) [static]
```

Parameters

<i>ptr</i>	
------------	--

Returns

6.222.3 Member Data Documentation

6.222.3.1 Ptr

```
NetworkObjectHeader* Ptr
```

Pointer to a [NetworkObjectHeader](#).

6.222.4 Property Documentation

6.222.4.1 Id

```
NetworkId Id [get]
```

Gets the Id of the [NetworkObjectHeader](#) this struct points to.

6.222.4.2 Type

```
NetworkObjectId Type [get]
```

Gets the Type of the [NetworkObjectHeader](#) this struct points to.

6.223 NetworkObjectInitializerUnity Class Reference

Initializes network objects for Unity.

Inherits [INetworkObjectInitializer](#).

Public Member Functions

- void [InitializeNetworkState](#) ([NetworkObject](#) networkObject)
Initializes the network object.

6.223.1 Detailed Description

Initializes network objects for Unity.

6.223.2 Member Function Documentation

6.223.2.1 InitializeNetworkState()

```
void InitializeNetworkState (  
    NetworkObject networkObject )
```

Initializes the network object.

Parameters

<i>networkObject</i>	The network object to initialize.
----------------------	-----------------------------------

Implements [INetworkObjectInitializer](#).

6.224 NetworkObjectMeta Class Reference

Meta information about a network object.

Properties

- [NetworkObjectHeaderFlags Flags](#) [get]
The flags indicating various states or properties of the network object.
- [NetworkId Id](#) [get]
Get the [NetworkId](#) of this object.
- [PlayerRef InputAuthority](#) [get]
Get the [Player](#) that has input authority over this object.
- [NetworkObjectNestingKey NestingKey](#) [get]
The nesting key of the network object.
- [NetworkId NestingRoot](#) [get]
Get the [NetworkId](#) of the root object in the nesting hierarchy.
- [PlayerRef StateAuthority](#) [get]
Get the [Player](#) that has state authority over this object.
- [NetworkObjectTypeId Type](#) [get]
Get the [NetworkObjectTypeId](#) of this object.

6.224.1 Detailed Description

Meta information about a network object.

6.224.2 Property Documentation

6.224.2.1 Flags

[NetworkObjectHeaderFlags](#) `Flags` [get]

The flags indicating various states or properties of the network object.

6.224.2.2 Id

[NetworkId](#) `Id` [get]

Get the [NetworkId](#) of this object.

6.224.2.3 InputAuthority

[PlayerRef](#) `InputAuthority` [get]

Get the [Player](#) that has input authority over this object.

6.224.2.4 NestingKey

`NetworkObjectNestingKey` NestingKey [get]

The nesting key of the network object.

6.224.2.5 NestingRoot

`NetworkId` NestingRoot [get]

Get the `NetworkId` of the root object in the nesting hierarchy.

6.224.2.6 StateAuthority

`PlayerRef` StateAuthority [get]

Get the Player that has state authority over this object.

6.224.2.7 Type

`NetworkObjectTypeId` Type [get]

Get the `NetworkObjectTypeId` of this object.

6.225 NetworkObjectNestingKey Struct Reference

A key used to identify a network object nesting.

Inherits `INetworkStruct`, and `IEquatable< NetworkObjectNestingKey >`.

Classes

- class `EqualityComparer`
Implements the `IEqualityComparer` interface.

Public Member Functions

- bool [Equals](#) ([NetworkObjectNestingKey](#) other)
Checks if the current instance of [NetworkObjectNestingKey](#) is equal to another instance of the same type.
- override bool [Equals](#) (object obj)
Checks if the current instance of [NetworkObjectNestingKey](#) is equal to another object.
- override int [GetHashCode](#) ()
Serves as the default hash function.
- [NetworkObjectNestingKey](#) (int value)
Initializes a new instance of the [NetworkObjectNestingKey](#) struct with a specified value.
- override string [ToString](#) ()
Returns a string that represents the current object.

Public Attributes

- int [Value](#)
The value of the [NetworkObjectNestingKey](#).

Static Public Attributes

- const int [ALIGNMENT](#) = 4
The alignment of the [NetworkObjectNestingKey](#) in bytes.
- const int [SIZE](#) = 4
The size of the [NetworkObjectNestingKey](#) in bytes.

Properties

- bool [IsNone](#) [get]
Checks if the [NetworkObjectNestingKey](#) is none.
- bool [IsValid](#) [get]
Checks if the [NetworkObjectNestingKey](#) is valid.

6.225.1 Detailed Description

A key used to identify a network object nesting.

6.225.2 Constructor & Destructor Documentation

6.225.2.1 [NetworkObjectNestingKey](#)()

```
NetworkObjectNestingKey (  
    int value )
```

Initializes a new instance of the [NetworkObjectNestingKey](#) struct with a specified value.

Parameters

<i>value</i>	The value of the NetworkObjectNestingKey .
--------------	--

6.225.3 Member Function Documentation

6.225.3.1 Equals() [1/2]

```
bool Equals (
    NetworkObjectNestingKey other )
```

Checks if the current instance of [NetworkObjectNestingKey](#) is equal to another instance of the same type.

Parameters

<i>other</i>	An instance of NetworkObjectNestingKey to compare with the current instance.
--------------	--

Returns

True if the current instance is equal to the other parameter; otherwise, false.

6.225.3.2 Equals() [2/2]

```
override bool Equals (
    object obj )
```

Checks if the current instance of [NetworkObjectNestingKey](#) is equal to another object.

Parameters

<i>obj</i>	An object to compare with the current instance.
------------	---

Returns

True if the current instance is equal to the obj parameter; otherwise, false.

6.225.3.3 GetHashCode()

```
override int GetHashCode ( )
```

Serves as the default hash function.

Returns

A hash code for the current object.

6.225.3.4 ToString()

```
override string ToString ( )
```

Returns a string that represents the current object.

Returns

A string that represents the current object.

6.225.4 Member Data Documentation**6.225.4.1 ALIGNMENT**

```
const int ALIGNMENT = 4 [static]
```

The alignment of the [NetworkObjectNestingKey](#) in bytes.

6.225.4.2 SIZE

```
const int SIZE = 4 [static]
```

The size of the [NetworkObjectNestingKey](#) in bytes.

6.225.4.3 Value

```
int Value
```

The value of the [NetworkObjectNestingKey](#).

6.225.5 Property Documentation

6.225.5.1 IsNone

```
bool IsNone [get]
```

Checks if the [NetworkObjectNestingKey](#) is none.

Returns

True if the value of the [NetworkObjectNestingKey](#) is 0; otherwise, false.

6.225.5.2 IsValid

```
bool IsValid [get]
```

Checks if the [NetworkObjectNestingKey](#) is valid.

Returns

True if the value of the [NetworkObjectNestingKey](#) is greater than 0; otherwise, false.

6.226 NetworkObjectNestingKey.EqualityComparer Class Reference

Implements the [IEqualityComparer](#) interface.

Inherits [IEqualityComparer< NetworkObjectNestingKey >](#).

Public Member Functions

- [bool Equals](#) ([NetworkObjectNestingKey](#) x, [NetworkObjectNestingKey](#) y)
Determines whether two [NetworkObjectNestingKey](#) objects are equal.
- [int GetHashCode](#) ([NetworkObjectNestingKey](#) obj)
Returns a hash code for the specified [NetworkObjectNestingKey](#).

6.226.1 Detailed Description

Implements the [IEqualityComparer](#) interface.

6.226.2 Member Function Documentation

6.226.2.1 Equals()

```
bool Equals (
    NetworkObjectNestingKey x,
    NetworkObjectNestingKey y )
```

Determines whether two [NetworkObjectNestingKey](#) objects are equal.

6.226.2.2 GetHashCode()

```
int GetHashCode (
    NetworkObjectNestingKey obj )
```

Returns a hash code for the specified [NetworkObjectNestingKey](#).

6.227 NetworkObjectPrefabData Class Reference

This class represents the data for a network object prefab.

Inherits [Behaviour](#).

Public Attributes

- [NetworkObjectGuid](#) `Guid`
The unique identifier for the network object.

Additional Inherited Members

6.227.1 Detailed Description

This class represents the data for a network object prefab.

6.227.2 Member Data Documentation

6.227.2.1 Guid

[NetworkObjectGuid](#) `Guid`

The unique identifier for the network object.

6.228 NetworkObjectProviderDummy Class Reference

A dummy implementation of the [INetworkObjectProvider](#) interface. This class is used for testing purposes and throws a `NotImplementedException` for all its methods.

Inherits [INetworkObjectProvider](#).

Public Member Functions

- [NetworkObjectAcquireResult AcquirePrefabInstance](#) ([NetworkRunner](#) runner, in [NetworkPrefabAcquireContext](#) context, out [NetworkObject](#) instance)
Acquires an instance of a prefab for a network object.
- [NetworkPrefabId GetPrefabId](#) ([NetworkRunner](#) runner, [NetworkObjectGuid](#) prefabGuid)
Translates guid into prefab id.
- void [ReleaseInstance](#) ([NetworkRunner](#) runner, in [NetworkObjectReleaseContext](#) context)
Releases an instance of a network object.

6.228.1 Detailed Description

A dummy implementation of the [INetworkObjectProvider](#) interface. This class is used for testing purposes and throws a `NotImplementedException` for all its methods.

6.229 NetworkObjectReleaseContext Struct Reference

Represents the context for releasing a network object. This struct is unsafe because it uses pointers.

Public Member Functions

- [NetworkObjectReleaseContext](#) ([NetworkObject](#) obj, [NetworkObjectTypeId](#) typeId, bool isBeingDestroyed, bool isNested)
Initializes a new instance of the [NetworkObjectReleaseContext](#) struct with the specified parameters.
- override string [ToString](#) ()
Returns a string that represents the current object.

Public Attributes

- readonly bool [IsBeingDestroyed](#)
Indicates whether the network object is being destroyed.
- readonly bool [IsNestedObject](#)
Indicates whether the network object is a nested object.
- readonly [NetworkObject](#) [Object](#)
The network object to be released.
- readonly [NetworkObjectTypeId](#) [TypeId](#)
The type identifier of the network object.

6.229.1 Detailed Description

Represents the context for releasing a network object. This struct is unsafe because it uses pointers.

6.229.2 Constructor & Destructor Documentation

6.229.2.1 NetworkObjectReleaseContext()

```
NetworkObjectReleaseContext (  
    NetworkObject obj,  
    NetworkObjectId typeId,  
    bool isBeingDestroyed,  
    bool isNested )
```

Initializes a new instance of the [NetworkObjectReleaseContext](#) struct with the specified parameters.

Parameters

<i>obj</i>	The network object to be released.
<i>typeId</i>	The type identifier of the network object.
<i>isBeingDestroyed</i>	Indicates whether the network object is being destroyed.
<i>isNested</i>	Indicates whether the network object is a nested object.

6.229.3 Member Function Documentation

6.229.3.1 ToString()

```
override string ToString ( )
```

Returns a string that represents the current object.

Returns

A string that represents the current object.

6.229.4 Member Data Documentation

6.229.4.1 IsBeingDestroyed

readonly bool IsBeingDestroyed

Indicates whether the network object is being destroyed.

6.229.4.2 IsNestedObject

readonly bool IsNestedObject

Indicates whether the network object is a nested object.

6.229.4.3 Object

readonly [NetworkObject](#) Object

The network object to be released.

6.229.4.4 TypeId

readonly [NetworkObjectId](#) TypeId

The type identifier of the network object.

6.230 NetworkObjectSortKeyComparer Class Reference

This class is used to compare two [NetworkObject](#) instances based on their SortKey. It implements the IComparer interface.

Inherits IComparer< [NetworkObject](#) >.

Public Member Functions

- int [Compare](#) ([NetworkObject](#) x, [NetworkObject](#) y)
Compares two [NetworkObject](#) instances based on their SortKey.

Static Public Attributes

- static readonly [NetworkObjectSortKeyComparer](#) Instance = new [NetworkObjectSortKeyComparer](#)()
An instance of the [NetworkObjectSortKeyComparer](#) class.

6.230.1 Detailed Description

This class is used to compare two [NetworkObject](#) instances based on their SortKey. It implements the IComparer interface.

6.230.2 Member Function Documentation

6.230.2.1 Compare()

```
int Compare (
    NetworkObject x,
    NetworkObject y )
```

Compares two [NetworkObject](#) instances based on their SortKey.

Parameters

<i>x</i>	The first NetworkObject to compare.
<i>y</i>	The second NetworkObject to compare.

Returns

A signed integer that indicates the relative values of x and y.

6.230.3 Member Data Documentation

6.230.3.1 Instance

```
readonly NetworkObjectSortKeyComparer Instance = new NetworkObjectSortKeyComparer() [static]
```

An instance of the [NetworkObjectSortKeyComparer](#) class.

6.231 NetworkObjectSpawnException Class Reference

Network Object Spawn Exception

Inherits Exception.

Public Member Functions

- [NetworkObjectSpawnException](#) ([NetworkSpawnStatus](#) status, [NetworkObjectTypeId?](#) id=null)
Network Object Spawn Exception Constructor

Properties

- override string [Message](#) [get]
Exception Message
- [NetworkSpawnStatus Status](#) [get]
Network Spawn Status
- [NetworkObjectId? TypeId](#) [get]
Network Object Type Id

6.231.1 Detailed Description

Network Object Spawn Exception

6.231.2 Constructor & Destructor Documentation

6.231.2.1 NetworkObjectSpawnException()

```
NetworkObjectSpawnException (  
    NetworkSpawnStatus status,  
    NetworkObjectId? id = null )
```

Network Object Spawn Exception Constructor

Parameters

<i>status</i>	Network Spawn Status
<i>id</i>	Network Object Type Id

6.231.3 Property Documentation

6.231.3.1 Message

```
override string Message [get]
```

Exception Message

6.231.3.2 Status

```
NetworkSpawnStatus Status [get]
```

Network Spawn Status

6.231.3.3 Typeld

`NetworkObjectTypeId?` `TypeId` [get]

Network Object Type Id

6.232 NetworkObjectTypeId Struct Reference

ID for a [NetworkObject](#) Prefab which has been cataloged in a [NetworkProjectConfig.PrefabTable](#).

Inherits [INetworkStruct](#), and [IEquatable< NetworkObjectTypeId >](#).

Classes

- class [EqualityComparer](#)
NetworkObjectTypeId Comparer

Public Member Functions

- bool [Equals](#) ([NetworkObjectTypeId](#) other)
Checks if the current [NetworkObjectTypeId](#) instance is equal to another [NetworkObjectTypeId](#) instance.
- override bool [Equals](#) (object obj)
Determines whether the specified object is equal to the current [NetworkObjectTypeId](#) instance.
- override int [GetHashCode](#) ()
Generates a hash code for the current [NetworkObjectTypeId](#) instance.
- override string [ToString](#) ()
Returns a string that represents the current [NetworkObjectTypeId](#) instance.

Static Public Member Functions

- static [NetworkObjectTypeId FromCustom](#) (uint raw)
Creates a [NetworkObjectTypeId](#) from a raw uint value representing a Custom type.
- static [NetworkObjectTypeId FromPrefabId](#) ([NetworkPrefabId](#) prefabId)
Creates a [NetworkObjectTypeId](#) from a [NetworkPrefabId](#).
- static [NetworkObjectTypeId FromSceneObjectId](#) ([NetworkSceneObjectId](#) sceneObjectId)
Creates a [NetworkObjectTypeId](#) from a [NetworkSceneObjectId](#).
- static [NetworkObjectTypeId FromSceneRefAndObjectIndex](#) ([SceneRef](#) sceneRef, int objIndex, [NetworkSceneLoadId](#) loadId=default)
Creates a [NetworkObjectTypeId](#) from a [SceneRef](#), an object index, and an optional [NetworkSceneLoadId](#).
- static [NetworkObjectTypeId FromStruct](#) (ushort structId)
Creates a [NetworkObjectTypeId](#) from a ushort value representing an InternalStruct type.
- static implicit operator [NetworkObjectTypeId](#) ([NetworkPrefabId](#) prefabId)
Converts a [NetworkPrefabId](#) instance to a [NetworkObjectTypeId](#) instance.
- static bool operator != ([NetworkObjectTypeId](#) a, [NetworkObjectTypeId](#) b)
Determines whether two [NetworkObjectTypeId](#) instances are not equal.
- static bool operator == ([NetworkObjectTypeId](#) a, [NetworkObjectTypeId](#) b)
Determines whether two [NetworkObjectTypeId](#) instances are equal.

Public Attributes

- uint `_value0`
Represents the first part of the value of a [NetworkObjectTypeId](#).
- uint `_value1`
Represents the second part of the value of a [NetworkObjectTypeId](#).

Static Public Attributes

- const int `ALIGNMENT` = 4
Represents the alignment of a [NetworkObjectTypeId](#) in memory.
- const int `MAX_SCENE_OBJECT_INDEX` = (1 << SCENE_OBJECT_INDEX_BITS) - 1
Represents the maximum number of SceneObjects that can be represented by a [NetworkObjectTypeId](#).
- const int `SIZE` = 8
Represents the size of a [NetworkObjectTypeId](#) in bytes.
- const ushort `STRUCT_TYPE_PLAYERDATA` = 1

Properties

- uint `AsCustom` [get]
Gets the raw uint value representation of the [NetworkObjectTypeId](#) assuming it is a Custom type.
- ushort `AsInternalStructId` [get]
Gets the ushort value representation of the [NetworkObjectTypeId](#) assuming it is an InternalStruct type.
- [NetworkPrefabId](#) `AsPrefabId` [get]
Gets the [NetworkPrefabId](#) representation of the [NetworkObjectTypeId](#) assuming it is a Prefab.
- [NetworkSceneObjectId](#) `AsSceneObjectId` [get]
Gets the [NetworkSceneObjectId](#) representation of the [NetworkObjectTypeId](#) assuming it is a SceneObject.
- static [EqualityComparer](#) `Comparer` = new [EqualityComparer](#)() [get]
An instance of the [NetworkObjectTypeId](#) [EqualityComparer](#) class.
- bool `IsCustom` [get]
Checks if the [NetworkObjectTypeId](#) is a Custom type.
- bool `IsNone` [get]
Checks if the [NetworkObjectTypeId](#) is invalid.
- bool `IsPrefab` [get]
Checks if the [NetworkObjectTypeId](#) is a Prefab.
- bool `IsSceneObject` [get]
Checks if the [NetworkObjectTypeId](#) is a SceneObject.
- bool `IsStruct` [get]
Checks if the [NetworkObjectTypeId](#) is an InternalStruct.
- bool `IsValid` [get]
Checks if the [NetworkObjectTypeId](#) is valid.
- [NetworkTypeIdKind](#) `Kind` [get]
Gets the kind of the [NetworkObjectTypeId](#).
- static [NetworkObjectTypeId](#) `PlayerData` [get]
Represents a [NetworkObjectTypeId](#) for the PlayerData.

6.232.1 Detailed Description

ID for a [NetworkObject](#) Prefab which has been cataloged in a [NetworkProjectConfig.PrefabTable](#).

6.232.2 Member Function Documentation

6.232.2.1 Equals() [1/2]

```
bool Equals (
    NetworkObjectId other )
```

Checks if the current [NetworkObjectId](#) instance is equal to another [NetworkObjectId](#) instance.

Parameters

<i>other</i>	The other NetworkObjectId instance to compare with the current instance.
--------------	--

6.232.2.2 Equals() [2/2]

```
override bool Equals (
    object obj )
```

Determines whether the specified object is equal to the current [NetworkObjectId](#) instance.

Parameters

<i>obj</i>	The object to compare with the current NetworkObjectId instance.
------------	--

6.232.2.3 FromCustom()

```
static NetworkObjectId FromCustom (
    uint raw ) [static]
```

Creates a [NetworkObjectId](#) from a raw uint value representing a Custom type.

Parameters

<i>raw</i>	The raw uint value to use.
------------	----------------------------

Returns

A [NetworkObjectId](#) that represents a Custom type with the given raw value.

6.232.2.4 FromPrefabId()

```
static NetworkObjectTypeId FromPrefabId (  
    NetworkPrefabId prefabId ) [static]
```

Creates a [NetworkObjectTypeId](#) from a [NetworkPrefabId](#).

Parameters

<i>prefabId</i>	The NetworkPrefabId to use.
-----------------	---

Returns

A [NetworkObjectTypeId](#) that represents a Prefab with the given [NetworkPrefabId](#).

Exceptions

<i>ArgumentException</i>	Thrown when the provided NetworkPrefabId is not valid.
--------------------------	--

6.232.2.5 FromSceneObjectId()

```
static NetworkObjectTypeId FromSceneObjectId (  
    NetworkSceneObjectId sceneObjectId ) [static]
```

Creates a [NetworkObjectTypeId](#) from a [NetworkSceneObjectId](#).

Parameters

<i>sceneObjectId</i>	
----------------------	--

Returns

6.232.2.6 FromSceneRefAndObjectIndex()

```
static NetworkObjectTypeId FromSceneRefAndObjectIndex (  
    SceneRef sceneRef,  
    int objIndex,  
    NetworkSceneLoadId loadId = default ) [static]
```

Creates a [NetworkObjectTypeId](#) from a [SceneRef](#), an object index, and an optional [NetworkSceneLoadId](#).

Parameters

<i>sceneRef</i>	The SceneRef to use.
<i>objIndex</i>	The object index to use.
<i>loadId</i>	The NetworkSceneLoadId to use. Defaults to default(NetworkSceneLoadId).

Returns

A [NetworkObjectTypeId](#) that represents a SceneObject with the given [SceneRef](#), object index, and [NetworkSceneLoadId](#).

Exceptions

<i>ArgumentException</i>	Thrown when the provided SceneRef is not valid.
<i>ArgumentOutOfRangeException</i>	Thrown when the provided object index is out of range.

6.232.2.7 FromStruct()

```
static NetworkObjectTypeId FromStruct (
    ushort structId ) [static]
```

Creates a [NetworkObjectTypeId](#) from a ushort value representing an InternalStruct type.

Parameters

<i>struct↔ Id</i>	The ushort value to use.
-----------------------	--------------------------

Returns

A [NetworkObjectTypeId](#) that represents an InternalStruct type with the given ushort value.

6.232.2.8 GetHashCode()

```
override int GetHashCode ( )
```

Generates a hash code for the current [NetworkObjectTypeId](#) instance.

6.232.2.9 operator NetworkObjectTypeId()

```
static implicit operator NetworkObjectTypeId (
    NetworkPrefabId prefabId ) [static]
```

Converts a [NetworkPrefabId](#) instance to a [NetworkObjectTypeId](#) instance.

Parameters

<i>prefab</i> ↔ <i>Id</i>	The NetworkPrefabId instance to convert.
------------------------------	--

Returns

A [NetworkObjectTypeId](#) instance that represents a Prefab with the given [NetworkPrefabId](#).

6.232.2.10 operator"!="()

```
static bool operator!= (
    NetworkObjectId a,
    NetworkObjectId b ) [static]
```

Determines whether two [NetworkObjectTypeId](#) instances are not equal.

6.232.2.11 operator=="()

```
static bool operator== (
    NetworkObjectId a,
    NetworkObjectId b ) [static]
```

Determines whether two [NetworkObjectTypeId](#) instances are equal.

6.232.2.12 ToString()

```
override string ToString ( )
```

Returns a string that represents the current [NetworkObjectTypeId](#) instance.

6.232.3 Member Data Documentation**6.232.3.1 _value0**

```
uint _value0
```

Represents the first part of the value of a [NetworkObjectTypeId](#).

6.232.3.2 `_value1`

```
uint _value1
```

Represents the second part of the value of a [NetworkObjectTypeId](#).

6.232.3.3 ALIGNMENT

```
const int ALIGNMENT = 4 [static]
```

Represents the alignment of a [NetworkObjectTypeId](#) in memory.

6.232.3.4 MAX_SCENE_OBJECT_INDEX

```
const int MAX_SCENE_OBJECT_INDEX = (1 << SCENE_OBJECT_INDEX_BITS) - 1 [static]
```

Represents the maximum number of SceneObjects that can be represented by a [NetworkObjectTypeId](#).

6.232.3.5 SIZE

```
const int SIZE = 8 [static]
```

Represents the size of a [NetworkObjectTypeId](#) in bytes.

6.232.4 Property Documentation

6.232.4.1 AsCustom

```
uint AsCustom [get]
```

Gets the raw uint value representation of the [NetworkObjectTypeId](#) assuming it is a Custom type.

The raw uint value representation of the [NetworkObjectTypeId](#).

Exceptions

<i>InvalidOperationException</i>	Thrown when the NetworkObjectTypeId is not a Custom type.
----------------------------------	---

6.232.4.2 AsInternalStructId

```
ushort AsInternalStructId [get]
```

Gets the ushort value representation of the [NetworkObjectTypeId](#) assuming it is an InternalStruct type.

The ushort value representation of the [NetworkObjectTypeId](#).

Exceptions

<i>InvalidOperationException</i>	Thrown when the NetworkObjectTypeId is not an InternalStruct type.
----------------------------------	--

6.232.4.3 AsPrefabId

```
NetworkPrefabId AsPrefabId [get]
```

Gets the [NetworkPrefabId](#) representation of the [NetworkObjectTypeId](#) assuming it is a Prefab.

The [NetworkPrefabId](#) representation of the [NetworkObjectTypeId](#).

Exceptions

<i>InvalidOperationException</i>	Thrown when the NetworkObjectTypeId is not a Prefab.
----------------------------------	--

6.232.4.4 AsSceneObjectId

```
NetworkSceneObjectId AsSceneObjectId [get]
```

Gets the [NetworkSceneObjectId](#) representation of the [NetworkObjectTypeId](#) assuming it is a SceneObject.

The [NetworkSceneObjectId](#) representation of the [NetworkObjectTypeId](#).

Exceptions

<i>InvalidOperationException</i>	Thrown when the NetworkObjectTypeId is not a SceneObject.
----------------------------------	---

6.232.4.5 Comparer

```
EqualityComparer Comparer = new EqualityComparer() [static], [get]
```

An instance of the [NetworkObjectTypeId EqualityComparer](#) class.

6.232.4.6 IsCustom

```
bool IsCustom [get]
```

Checks if the [NetworkObjectTypeId](#) is a Custom type.

6.232.4.7 IsNone

```
bool IsNone [get]
```

Checks if the [NetworkObjectTypeId](#) is invalid.

6.232.4.8 IsPrefab

```
bool IsPrefab [get]
```

Checks if the [NetworkObjectTypeId](#) is a Prefab.

6.232.4.9 IsSceneObject

```
bool IsSceneObject [get]
```

Checks if the [NetworkObjectTypeId](#) is a SceneObject.

6.232.4.10 IsStruct

```
bool IsStruct [get]
```

Checks if the [NetworkObjectTypeId](#) is an InternalStruct.

6.232.4.11 IsValid

```
bool IsValid [get]
```

Checks if the [NetworkObjectTypeId](#) is valid.

6.232.4.12 Kind

`NetworkTypeIdKind Kind [get]`

Gets the kind of the [NetworkObjectTypeId](#).

6.232.4.13 PlayerData

`NetworkObjectTypeId PlayerData [static], [get]`

Represents a [NetworkObjectTypeId](#) for the PlayerData.

6.233 NetworkObjectTypeId.EqualityComparer Class Reference

[NetworkObjectTypeId](#) Comparer

Inherits `IEqualityComparer< NetworkObjectTypeId >`.

Public Member Functions

- `bool Equals (NetworkObjectTypeId x, NetworkObjectTypeId y)`
Checks if two [NetworkObjectTypeId](#) instances are equal.
- `int GetHashCode (NetworkObjectTypeId obj)`
Gets the hash code of a [NetworkObjectTypeId](#) instance.

6.233.1 Detailed Description

[NetworkObjectTypeId](#) Comparer

6.233.2 Member Function Documentation

6.233.2.1 Equals()

```
bool Equals (  
    NetworkObjectTypeId x,  
    NetworkObjectTypeId y )
```

Checks if two [NetworkObjectTypeId](#) instances are equal.

6.233.2.2 GetHashCode()

```
int GetHashCode (  
    NetworkObjectTypeId obj )
```

Gets the hash code of a [NetworkObjectTypeId](#) instance.

Parameters

<i>obj</i>	The NetworkObjectTyped instance.
------------	--

6.234 NetworkPhysicsInfo Struct Reference

Network Physics [INetworkStruct](#)

Inherits [INetworkStruct](#).

Public Attributes

- float [TimeScale](#)
NetworkPhysicsInfo Time Scale

Static Public Attributes

- const int [SIZE](#) = 40
NetworkPhysicsInfo Total Size
- const int [WORD_COUNT](#) = 10
NetworkPhysicsInfo Word Count

6.234.1 Detailed Description

Network Physics [INetworkStruct](#)

6.234.2 Member Data Documentation

6.234.2.1 SIZE

```
const int SIZE = 40 [static]
```

[NetworkPhysicsInfo](#) Total Size

6.234.2.2 TimeScale

```
float TimeScale
```

[NetworkPhysicsInfo](#) Time Scale

6.234.2.3 WORD_COUNT

```
const int WORD_COUNT = 10 [static]
```

[NetworkPhysicsInfo](#) Word Count

6.235 NetworkPrefabAcquireContext Struct Reference

Represents the context for acquiring a prefab instance for a network object. This struct is unsafe because it uses pointers.

Public Member Functions

- [NetworkPrefabAcquireContext](#) ([NetworkPrefabId](#) prefabId, [NetworkObjectMeta](#) meta=null, bool isSynchronous=true, bool dontDestroyOnLoad=false)

Initializes a new instance of the [NetworkPrefabAcquireContext](#) struct with the specified parameters.

Public Attributes

- readonly bool [DontDestroyOnLoad](#)
Indicates whether the network object should not be destroyed on load.
- readonly bool [IsSynchronous](#)
Indicates whether the operation is synchronous.
- readonly [NetworkObjectMeta](#) [Meta](#)
The metadata of the network object.
- readonly [NetworkPrefabId](#) [PrefabId](#)
The identifier of the prefab.

Properties

- int * [Data](#) [get]
Gets the data pointer to the first word of this [NetworkObject](#)'s data block.
- bool [HasHeader](#) [get]
Checks if the Header is not null.

6.235.1 Detailed Description

Represents the context for acquiring a prefab instance for a network object. This struct is unsafe because it uses pointers.

6.235.2 Constructor & Destructor Documentation

6.235.2.1 NetworkPrefabAcquireContext()

```
NetworkPrefabAcquireContext (
    NetworkPrefabId prefabId,
    NetworkObjectMeta meta = null,
    bool isSynchronous = true,
    bool dontDestroyOnLoad = false )
```

Initializes a new instance of the [NetworkPrefabAcquireContext](#) struct with the specified parameters.

Parameters

<i>prefabId</i>	The identifier of the prefab.
<i>meta</i>	The metadata of the network object.
<i>isSynchronous</i>	Indicates whether the operation is synchronous.
<i>dontDestroyOnLoad</i>	Indicates whether the network object should not be destroyed on load.

6.235.3 Member Data Documentation

6.235.3.1 DontDestroyOnLoad

readonly bool DontDestroyOnLoad

Indicates whether the network object should not be destroyed on load.

6.235.3.2 IsSynchronous

readonly bool IsSynchronous

Indicates whether the operation is synchronous.

6.235.3.3 Meta

readonly [NetworkObjectMeta](#) Meta

The metadata of the network object.

6.235.3.4 PrefabId

readonly [NetworkPrefabId](#) PrefabId

The identifier of the prefab.

6.235.4 Property Documentation

6.235.4.1 Data

int* Data [get]

Gets the data pointer to the first word of this [NetworkObject](#)'s data block.

Returns

Data pointer to the first word of this [NetworkObject](#)'s data block.

Exceptions

<code>InvalidOperationException</code>	Thrown when the Header is null.
--	---------------------------------

6.235.4.2 HasHeader

```
bool HasHeader [get]
```

Checks if the Header is not null.

Returns

True if the Header is not null; otherwise, false.

6.236 NetworkPrefabAttribute Class Reference

Network Prefab Attribute

Inherits [PropertyAttribute](#).

6.236.1 Detailed Description

Network Prefab Attribute

6.237 NetworkPrefabId Struct Reference

ID for a [NetworkObject](#) Prefab which has been cataloged in a [NetworkProjectConfig.PrefabTable](#).

Inherits [INetworkStruct](#), [IEquatable< NetworkPrefabId >](#), [IComparable](#), and [IComparable< NetworkPrefabId >](#).

Classes

- class [EqualityComparer](#)
Equality comparer for [NetworkPrefabId](#).

Public Member Functions

- int [CompareTo](#) ([NetworkPrefabId](#) other)
Compares the [NetworkPrefabId](#) to another [NetworkPrefabId](#).
- int IComparable. [CompareTo](#) (object obj)
Compares the [NetworkPrefabId](#) to another object.
- bool [Equals](#) ([NetworkPrefabId](#) other)
Checks if the [NetworkPrefabId](#) is equal to another [NetworkPrefabId](#).
- override bool [Equals](#) (object obj)
Checks if the [NetworkPrefabId](#) is equal to another object.
- override int [GetHashCode](#) ()
Gets the hash code of the [NetworkPrefabId](#).
- override string [ToString](#) ()
Converts the [NetworkPrefabId](#) to a string.
- string [ToString](#) (bool brackets, bool prefix)
Converts the [NetworkPrefabId](#) to a string with optional brackets and prefix.

Static Public Member Functions

- static [NetworkPrefabId](#) [FromIndex](#) (int index)
Creates a [NetworkPrefabId](#) from an index.
- static [NetworkPrefabId](#) [FromRaw](#) (uint value)
Creates a [NetworkPrefabId](#) from a raw value.
- static bool [operator!=](#) ([NetworkPrefabId](#) a, [NetworkPrefabId](#) b)
Checks if two [NetworkPrefabId](#) are not equal.
- static bool [operator==](#) ([NetworkPrefabId](#) a, [NetworkPrefabId](#) b)
Checks if two [NetworkPrefabId](#) are equal.

Public Attributes

- uint [RawValue](#)
The raw value of the [NetworkPrefabId](#).

Static Public Attributes

- const int [ALIGNMENT](#) = 4
The alignment of a [NetworkPrefabId](#).
- const int [MAX_INDEX](#) = int.MaxValue - 1
The maximum index value of a [NetworkPrefabId](#).
- const int [SIZE](#) = 4
The size of a [NetworkPrefabId](#).

Properties

- int [AsIndex](#) [get]
Converts the [NetworkPrefabId](#) to an index.
- bool [IsNone](#) [get]
Checks if the [NetworkPrefabId](#) is none.
- bool [IsValid](#) [get]
Checks if the [NetworkPrefabId](#) is valid.

6.237.1 Detailed Description

ID for a [NetworkObject](#) Prefab which has been cataloged in a [NetworkProjectConfig.PrefabTable](#).

6.237.2 Member Function Documentation

6.237.2.1 CompareTo() [1/2]

```
int CompareTo (  
    NetworkPrefabId other )
```

Compares the [NetworkPrefabId](#) to another [NetworkPrefabId](#).

6.237.2.2 CompareTo() [2/2]

```
int IComparable. CompareTo (  
    object obj )
```

Compares the [NetworkPrefabId](#) to another object.

6.237.2.3 Equals() [1/2]

```
bool Equals (  
    NetworkPrefabId other )
```

Checks if the [NetworkPrefabId](#) is equal to another [NetworkPrefabId](#).

6.237.2.4 Equals() [2/2]

```
override bool Equals (  
    object obj )
```

Checks if the [NetworkPrefabId](#) is equal to another object.

6.237.2.5 FromIndex()

```
static NetworkPrefabId FromIndex (
    int index ) [static]
```

Creates a [NetworkPrefabId](#) from an index.

6.237.2.6 FromRaw()

```
static NetworkPrefabId FromRaw (
    uint value ) [static]
```

Creates a [NetworkPrefabId](#) from a raw value.

6.237.2.7 GetHashCode()

```
override int GetHashCode ( )
```

Gets the hash code of the [NetworkPrefabId](#).

6.237.2.8 operator"!="()

```
static bool operator!= (
    NetworkPrefabId a,
    NetworkPrefabId b ) [static]
```

Checks if two [NetworkPrefabId](#) are not equal.

6.237.2.9 operator=="()

```
static bool operator== (
    NetworkPrefabId a,
    NetworkPrefabId b ) [static]
```

Checks if two [NetworkPrefabId](#) are equal.

6.237.2.10 ToString() [1/2]

```
override string ToString ( )
```

Converts the [NetworkPrefabId](#) to a string.

6.237.2.11 ToString() [2/2]

```
string ToString (
    bool brackets,
    bool prefix )
```

Converts the [NetworkPrefabId](#) to a string with optional brackets and prefix.

6.237.3 Member Data Documentation

6.237.3.1 ALIGNMENT

```
const int ALIGNMENT = 4 [static]
```

The alignment of a [NetworkPrefabId](#).

6.237.3.2 MAX_INDEX

```
const int MAX_INDEX = int.MaxValue - 1 [static]
```

The maximum index value of a [NetworkPrefabId](#).

6.237.3.3 RawValue

```
uint RawValue
```

The raw value of the [NetworkPrefabId](#).

6.237.3.4 SIZE

```
const int SIZE = 4 [static]
```

The size of a [NetworkPrefabId](#).

6.237.4 Property Documentation

6.237.4.1 AsIndex

```
int AsIndex [get]
```

Converts the [NetworkPrefabId](#) to an index.

6.237.4.2 IsNone

```
bool IsNone [get]
```

Checks if the [NetworkPrefabId](#) is none.

6.237.4.3 IsValid

```
bool IsValid [get]
```

Checks if the [NetworkPrefabId](#) is valid.

6.238 NetworkPrefabId.EqualityComparer Class Reference

Equality comparer for [NetworkPrefabId](#).

Inherits `IEqualityComparer< NetworkPrefabId >`.

Public Member Functions

- `bool Equals (NetworkPrefabId x, NetworkPrefabId y)`
Checks if two [NetworkPrefabId](#) are equal.
- `int GetHashCode (NetworkPrefabId obj)`
Gets the hash code of a [NetworkPrefabId](#).

6.238.1 Detailed Description

Equality comparer for [NetworkPrefabId](#).

6.238.2 Member Function Documentation

6.238.2.1 Equals()

```
bool Equals (
    NetworkPrefabId x,
    NetworkPrefabId y )
```

Checks if two [NetworkPrefabId](#) are equal.

6.238.2.2 GetHashCode()

```
int GetHashCode (
    NetworkPrefabId obj )
```

Gets the hash code of a [NetworkPrefabId](#).

6.239 NetworkPrefabInfo Struct Reference

Meta data for a [NetworkObject](#) prefab which has been cataloged in a [NetworkProjectConfig.PrefabTable](#).

Public Attributes

- readonly [NetworkObjectHeader](#) * [Header](#)
Header data for the [NetworkObject](#) prefab.
- readonly bool [IsSynchronous](#)
Is the prefab supposed to be loaded in a synchronous way. [Fusion](#) will report an error if this field is set to true and no prefab is returned by [INetworkObjectProvider](#).
- readonly [NetworkPrefabId](#) [Prefab](#)
Prefab ID. Use [NetworkPrefabTable.TryAddSource](#) to look up the actual prefab reference in the [NetworkProjectConfig.PrefabTable](#).

Properties

- int * [Data](#) [get]
Data pointer to the first word of this [NetworkObject](#)'s data block.
- bool [HasHeader](#) [get]
If the Header is not null.

6.239.1 Detailed Description

Meta data for a [NetworkObject](#) prefab which has been cataloged in a [NetworkProjectConfig.PrefabTable](#).

6.239.2 Member Data Documentation

6.239.2.1 Header

```
readonly NetworkObjectHeader\* Header
```

Header data for the [NetworkObject](#) prefab.

6.239.2.2 IsSynchronous

```
readonly bool IsSynchronous
```

Is the prefab supposed to be loaded in a synchronous way. [Fusion](#) will report an error if this field is set to true and no prefab is returned by [INetworkObjectProvider](#).

6.239.2.3 Prefab

```
readonly NetworkPrefabId Prefab
```

Prefab ID. Use [NetworkPrefabTable.TryAddSource](#) to look up the actual prefab reference in the [NetworkProjectConfig.PrefabTable](#).

6.239.3 Property Documentation

6.239.3.1 Data

```
int* Data [get]
```

Data pointer to the first word of this [NetworkObject](#)'s data block.

6.239.3.2 HasHeader

```
bool HasHeader [get]
```

If the Header is not null.

6.240 NetworkPrefabRef Struct Reference

NetworkPrefabRef

Inherits [INetworkStruct](#), [IEquatable< NetworkPrefabRef >](#), and [IComparable< NetworkPrefabRef >](#).

Classes

- class [EqualityComparer](#)
EqualityComparer for NetworkPrefabRef

Public Member Functions

- int [CompareTo](#) ([NetworkPrefabRef](#) other)
Compare the NetworkPrefabRef to another NetworkPrefabRef
- bool [Equals](#) ([NetworkPrefabRef](#) other)
Check if the NetworkPrefabRef is equal to another NetworkPrefabRef
- override bool [Equals](#) (object obj)
Check if the NetworkPrefabRef is equal to another object
- override int [GetHashCode](#) ()
Get the hashcode for a NetworkPrefabRef
- [NetworkPrefabRef](#) (byte *guid)
*Create a NetworkPrefabRef from a byte**
- [NetworkPrefabRef](#) (byte[] guid)
Create a NetworkPrefabRef from a byte array
- [NetworkPrefabRef](#) (long data0, long data1)
Create a new NetworkPrefabRef
- [NetworkPrefabRef](#) (string guid)
Create a new NetworkPrefabRef
- override string [ToString](#) ()
Returns a string representation of the NetworkPrefabRef.
- string [ToString](#) (string format)
Returns a string representation of the NetworkPrefabRef.
- string [ToUnityGuidString](#) ()
Returns a string representation of the NetworkPrefabRef.

Static Public Member Functions

- static implicit operator [Guid](#) ([NetworkPrefabRef](#) guid)
Implicit conversion from NetworkPrefabRef to Guid
- static operator [NetworkObjectGuid](#) ([NetworkPrefabRef](#) t)
Explicit conversion from NetworkPrefabRef to NetworkObjectGuid
- static implicit operator [NetworkPrefabRef](#) ([Guid](#) guid)
Implicit conversion from Guid to NetworkPrefabRef
- static bool operator [!=](#) ([NetworkPrefabRef](#) a, [NetworkPrefabRef](#) b)
Compare two NetworkPrefabRef
- static bool operator [==](#) ([NetworkPrefabRef](#) a, [NetworkPrefabRef](#) b)
Compare two NetworkPrefabRef
- static [NetworkPrefabRef](#) [Parse](#) (string str)
Parse a NetworkPrefabRef from a string.
- static bool [TryParse](#) (string str, out [NetworkPrefabRef](#) guid)
Try to parse a string into a NetworkPrefabRef

Public Attributes

- fixed long [RawGuidValue](#) [2]
The Raw Guid Value of the [NetworkPrefabRef](#)

Static Public Attributes

- const int [ALIGNMENT](#) = 4
The alignment of the [NetworkPrefabRef](#)
- const int [SIZE](#) = 16
The Size of the [NetworkPrefabRef](#) in bytes

Properties

- static [NetworkPrefabRef Empty](#) [get]
The default value of a [NetworkPrefabRef](#)
- bool [IsValid](#) [get]
Signal if the [NetworkPrefabRef](#) is valid.

6.240.1 Detailed Description

[NetworkPrefabRef](#)

A decoupled [NetworkObject](#) prefab reference. Internally stored as a GUID.

6.240.2 Constructor & Destructor Documentation

6.240.2.1 [NetworkPrefabRef\(\)](#) [1/4]

```
NetworkPrefabRef (
    string guid )
```

Create a new [NetworkPrefabRef](#)

Parameters

<i>guid</i>	The guid to use
-------------	-----------------

6.240.2.2 [NetworkPrefabRef\(\)](#) [2/4]

```
NetworkPrefabRef (
```

```
long data0,  
long data1 )
```

Create a new [NetworkPrefabRef](#)

Parameters

<i>data0</i>	Data0 of the Guid
<i>data1</i>	Data1 of the Guid

6.240.2.3 NetworkPrefabRef() [3/4]

```
NetworkPrefabRef (  
    byte[] guid )
```

Create a [NetworkPrefabRef](#) from a byte array

Parameters

<i>guid</i>	The byte array to create the NetworkPrefabRef from
-------------	--

6.240.2.4 NetworkPrefabRef() [4/4]

```
NetworkPrefabRef (  
    byte * guid )
```

Create a [NetworkPrefabRef](#) from a byte*

Parameters

<i>guid</i>	The byte* to create the NetworkPrefabRef from
-------------	---

6.240.3 Member Function Documentation

6.240.3.1 CompareTo()

```
int CompareTo (  
    NetworkPrefabRef other )
```

Compare the [NetworkPrefabRef](#) to another [NetworkPrefabRef](#)

Parameters

<i>other</i>	The other NetworkPrefabRef to compare against
--------------	---

Returns

0 if the [NetworkPrefabRef](#) are equal, -1 if this [NetworkPrefabRef](#) is less than the other, 1 if this [NetworkPrefabRef](#) is greater than the other

6.240.3.2 Equals() [1/2]

```
bool Equals (
    NetworkPrefabRef other )
```

Check if the [NetworkPrefabRef](#) is equal to another [NetworkPrefabRef](#)

Parameters

<i>other</i>	The other NetworkPrefabRef to check against
--------------	---

Returns

True if the [NetworkPrefabRefs](#) are equal, false otherwise

6.240.3.3 Equals() [2/2]

```
override bool Equals (
    object obj )
```

Check if the [NetworkPrefabRef](#) is equal to another object

Parameters

<i>obj</i>	The other object to check against
------------	-----------------------------------

Returns

True if the objects are equal, false otherwise

6.240.3.4 GetHashCode()

```
override int GetHashCode ( )
```

Get the hashcode for a [NetworkPrefabRef](#)

6.240.3.5 operator Guid()

```
static implicit operator Guid (  
    NetworkPrefabRef guid ) [static]
```

Implicit conversion from [NetworkPrefabRef](#) to Guid

Parameters

<i>guid</i>	NetworkPrefabRef to convert from
-------------	--

Returns

Guid

6.240.3.6 operator NetworkObjectGuid()

```
static operator NetworkObjectGuid (  
    NetworkPrefabRef t ) [explicit], [static]
```

Explicit conversion from [NetworkPrefabRef](#) to [NetworkObjectGuid](#)

Parameters

<i>t</i>	NetworkPrefabRef to convert from
----------	--

Returns

[NetworkObjectGuid](#)

6.240.3.7 operator NetworkPrefabRef()

```
static implicit operator NetworkPrefabRef (  
    Guid guid ) [static]
```

Implicit conversion from Guid to [NetworkPrefabRef](#)

Parameters

<i>guid</i>	Guid to convert from
-------------	----------------------

Returns

[NetworkPrefabRef](#)

6.240.3.8 operator"!="()

```
static bool operator!= (
    NetworkPrefabRef a,
    NetworkPrefabRef b ) [static]
```

Compare two [NetworkPrefabRef](#)

Returns

True if the [NetworkPrefabRef](#) are not equal, false otherwise

6.240.3.9 operator=="()

```
static bool operator== (
    NetworkPrefabRef a,
    NetworkPrefabRef b ) [static]
```

Compare two [NetworkPrefabRef](#)

Returns

True if the [NetworkPrefabRef](#) are equal, false otherwise

6.240.3.10 Parse()

```
static NetworkPrefabRef Parse (
    string str ) [static]
```

Parse a [NetworkPrefabRef](#) from a string.

Parameters

<i>str</i>	The string to parse.
------------	----------------------

Returns

The parsed [NetworkPrefabRef](#).

6.240.3.11 ToString() [1/2]

```
override string ToString ( )
```

Returns a string representation of the [NetworkPrefabRef](#).

6.240.3.12 ToString() [2/2]

```
string ToString (
    string format )
```

Returns a string representation of the [NetworkPrefabRef](#).

6.240.3.13 ToUnityGuidString()

```
string ToUnityGuidString ( )
```

Returns a string representation of the [NetworkPrefabRef](#).

6.240.3.14 TryParse()

```
static bool TryParse (
    string str,
    out NetworkPrefabRef guid ) [static]
```

Try to parse a string into a [NetworkPrefabRef](#)

Parameters

<i>str</i>	String to parse
<i>guid</i>	Parsed NetworkPrefabRef

Returns

True if the string was parsed successfully, false otherwise

6.240.4 Member Data Documentation

6.240.4.1 ALIGNMENT

```
const int ALIGNMENT = 4 [static]
```

The alignment of the [NetworkPrefabRef](#)

6.240.4.2 RawGuidValue

```
fixed long RawGuidValue[2]
```

The Raw Guid Value of the [NetworkPrefabRef](#)

6.240.4.3 SIZE

```
const int SIZE = 16 [static]
```

The Size of the [NetworkPrefabRef](#) in bytes

6.240.5 Property Documentation

6.240.5.1 Empty

```
NetworkPrefabRef Empty [static], [get]
```

The default value of a [NetworkPrefabRef](#)

6.240.5.2 IsValid

```
bool IsValid [get]
```

Signal if the [NetworkPrefabRef](#) is valid.

6.241 NetworkPrefabRef.EqualityComparer Class Reference

[EqualityComparer](#) for [NetworkPrefabRef](#)

Inherits [IEqualityComparer](#)< [NetworkPrefabRef](#) >.

Public Member Functions

- bool [Equals](#) ([NetworkPrefabRef](#) x, [NetworkPrefabRef](#) y)
Check if two [NetworkPrefabRef](#) are equals
- int [GetHashCode](#) ([NetworkPrefabRef](#) obj)
Get the hashcode for a [NetworkPrefabRef](#)

6.241.1 Detailed Description

[EqualityComparer](#) for [NetworkPrefabRef](#)

6.241.2 Member Function Documentation

6.241.2.1 Equals()

```
bool Equals (  
    NetworkPrefabRef x,  
    NetworkPrefabRef y )
```

Check if two [NetworkPrefabRef](#) are equals

6.241.2.2 GetHashCode()

```
int GetHashCode (  
    NetworkPrefabRef obj )
```

Get the hashcode for a [NetworkPrefabRef](#)

6.242 NetworkPrefabTable Class Reference

Class representing a table of network prefabs.

Public Member Functions

- int [AddInstance](#) ([NetworkPrefabId](#) prefabId)
Add an instance of a prefab id.
- [NetworkPrefabId](#) [AddSource](#) ([INetworkPrefabSource](#) source)
Adds a prefab source to the table.
- void [Clear](#) ()
Clear the prefab table.
- bool [Contains](#) ([NetworkPrefabId](#) prefabId)
Returns true if the prefab table contains a prefab with the given id.
- [IEnumerable](#)<[NetworkPrefabId](#), [INetworkPrefabSource](#)> [GetEntries](#) ()
Returns all entries in the table.
- [NetworkObjectGuid](#) [GetGuid](#) ([NetworkPrefabId](#) prefabId)
Gets a prefab guid by id.
- [NetworkPrefabId](#) [GetId](#) ([NetworkObjectGuid](#) guid)
Gets a prefab id by guid.
- int [GetInstanceCount](#) ([NetworkPrefabId](#) prefabId)
Get the instance count of a prefab id.
- [INetworkPrefabSource](#) [GetSource](#) ([NetworkObjectGuid](#) guid)
Gets a prefab source by guid.
- [INetworkPrefabSource](#) [GetSource](#) ([NetworkPrefabId](#) prefabId)
Gets a prefab source by id.
- bool [IsAcquired](#) ([NetworkPrefabId](#) prefabId)
Signal if a prefab id has been acquired.
- [NetworkObject](#) [Load](#) ([NetworkPrefabId](#) prefabId, bool isSynchronous)
Load a prefab by id.
- int [RemoveInstance](#) ([NetworkPrefabId](#) prefabId)
Remove an instance of a prefab id.
- bool [TryAddSource](#) ([INetworkPrefabSource](#) source, out [NetworkPrefabId](#) id)
Tries to add a prefab source to the table.
- bool [Unload](#) ([NetworkPrefabId](#) prefabId)
Unload a prefab by id.
- void [UnloadAll](#) ()
Unload all prefabs.
- int [UnloadUnreferenced](#) (bool includeIncompleteLoads=false)
Unload all unreferenced prefabs.

Public Attributes

- [NetworkPrefabTableOptions](#) [Options](#) = [NetworkPrefabTableOptions.Default](#)
Options for the [NetworkPrefabTable](#).

Properties

- [IReadOnlyList](#)< [INetworkPrefabSource](#) > [Prefabs](#) [get]
All prefab sources.
- int [Version](#) [get]
Prefab table version. Incremented every time a change occurs.

6.242.1 Detailed Description

Class representing a table of network prefabs.

6.242.2 Member Function Documentation

6.242.2.1 AddInstance()

```
int AddInstance (
    NetworkPrefabId prefabId )
```

Add an instance of a prefab id.

Parameters

<i>prefabId</i>	Id of the prefab.
-----------------	-------------------

Returns

The new instance count, or 0 if not found.

6.242.2.2 AddSource()

```
NetworkPrefabId AddSource (
    INetworkPrefabSource source )
```

Adds a prefab source to the table.

Parameters

<i>source</i>	Prefab source to add.
---------------	-----------------------

Exceptions

<i>ArgumentException</i>	Thrown if a prefab source with the same guid already exists.
--------------------------	--

6.242.2.3 Clear()

```
void Clear ( )
```

Clear the prefab table.

6.242.2.4 Contains()

```
bool Contains (
    NetworkPrefabId prefabId )
```

Returns true if the prefab table contains a prefab with the given id.

Parameters

<i>prefab</i> ↔ <i>Id</i>	Id of the prefab.
------------------------------	-------------------

Returns

True if the prefab table contains a prefab with the given id.

6.242.2.5 GetEntries()

```
IEnumerable<(NetworkPrefabId, INetworkPrefabSource)> GetEntries ( )
```

Returns all entries in the table.

6.242.2.6 GetGuid()

```
NetworkObjectGuid GetGuid (
    NetworkPrefabId prefabId )
```

Gets a prefab guid by id.

Parameters

<i>prefab</i> ↔ <i>Id</i>	Id of the prefab source.
------------------------------	--------------------------

Returns

The prefab guid, or default if not found.

6.242.2.7 GetId()

```
NetworkPrefabId GetId (
    NetworkObjectGuid guid )
```

Gets a prefab id by guid.

Parameters

<i>guid</i>	Guid of the prefab source.
-------------	----------------------------

Returns

The prefab id, or default if not found.

6.242.2.8 GetInstancesCount()

```
int GetInstancesCount (
    NetworkPrefabId prefabId )
```

Get the instance count of a prefab id.

Parameters

<i>prefabId</i>	Id of the prefab.
-----------------	-------------------

Returns

The instance count, or 0 if not found.

6.242.2.9 GetSource() [1/2]

```
INetworkPrefabSource GetSource (
    NetworkObjectGuid guid )
```

Gets a prefab source by guid.

Parameters

<i>guid</i>	Guid of the prefab source.
-------------	----------------------------

Returns

The prefab source, or default if not found.

6.242.2.10 GetSource() [2/2]

```
INetworkPrefabSource GetSource (
    NetworkPrefabId prefabId )
```

Gets a prefab source by id.

Parameters

<i>prefabId</i>	Id of the prefab source.
-----------------	--------------------------

Returns

The prefab source, or default if not found.

6.242.2.11 IsAcquired()

```
bool IsAcquired (
    NetworkPrefabId prefabId )
```

Signal if a prefab id has been acquired.

Parameters

<i>prefabId</i>	Id of the prefab.
-----------------	-------------------

Returns

True if the prefab id has been acquired.

6.242.2.12 Load()

```
NetworkObject Load (
    NetworkPrefabId prefabId,
    bool isSynchronous )
```

Load a prefab by id.

Parameters

<i>prefabId</i>	Id of the prefab.
<i>isSynchronous</i>	If true, the load will be synchronous.

Returns

The loaded prefab, or null if not found.

6.242.2.13 RemoveInstance()

```
int RemoveInstance (
    NetworkPrefabId prefabId )
```

Remove an instance of a prefab id.

Parameters

<i>prefabId</i> <i>id</i>	Id of the prefab.
------------------------------	-------------------

Returns

The new instance count, or 0 if not found.

6.242.2.14 TryAddSource()

```
bool TryAddSource (
    INetworkPrefabSource source,
    out NetworkPrefabId id )
```

Tries to add a prefab source to the table.

Parameters

<i>source</i>	Prefab source to add.
<i>id</i>	Id of the prefab source.

Returns

True if the prefab source was added, false otherwise.

Exceptions

<i>ArgumentNullException</i>	Thrown if <i>source</i> is null.
------------------------------	----------------------------------

6.242.2.15 Unload()

```
bool Unload (
    NetworkPrefabId prefabId )
```

Unload a prefab by id.

Parameters

<i>prefab</i> ↔ <i>Id</i>	Id of the prefab.
------------------------------	-------------------

Returns

True if the prefab was unloaded, false otherwise.

6.242.2.16 UnloadAll()

```
void UnloadAll ( )
```

Unload all prefabs.

6.242.2.17 UnloadUnreferenced()

```
int UnloadUnreferenced (
    bool includeIncompleteLoads = false )
```

Unload all unreferenced prefabs.

Parameters

<i>includeIncompleteLoads</i>	If true, incomplete loads will be unloaded as well.
-------------------------------	---

Returns

The number of prefabs unloaded.

6.242.3 Member Data Documentation

6.242.3.1 Options

```
NetworkPrefabTableOptions Options = NetworkPrefabTableOptions.Default
```

Options for the [NetworkPrefabTable](#).

6.242.4 Property Documentation

6.242.4.1 Prefabs

```
ReadOnlyList<INetworkPrefabSource> Prefabs [get]
```

All prefab sources.

6.242.4.2 Version

```
int Version [get]
```

Prefab table version. Incremented every time a change occurs.

6.243 NetworkPrefabTableOptions Struct Reference

Options for the [NetworkPrefabTable](#).

Public Attributes

- bool [UnloadPrefabOnReleasingLastInstance](#)
If true, prefabs will be unloaded when the last instance is released.
- bool [UnloadUnusedPrefabsOnShutdown](#)
If true, all prefabs will be unloaded on shutdown.

Static Public Attributes

- static [NetworkPrefabTableOptions Default](#)
Default options.

6.243.1 Detailed Description

Options for the [NetworkPrefabTable](#).

6.243.2 Member Data Documentation

6.243.2.1 Default

```
NetworkPrefabTableOptions Default [static]
```

Initial value:

```
= new NetworkPrefabTableOptions() {  
    UnloadPrefabOnReleasingLastInstance = false,  
    UnloadUnusedPrefabsOnShutdown = true,  
}
```

Default options.

6.243.2.2 UnloadPrefabOnReleasingLastInstance

```
bool UnloadPrefabOnReleasingLastInstance
```

If true, prefabs will be unloaded when the last instance is released.

6.243.2.3 UnloadUnusedPrefabsOnShutdown

```
bool UnloadUnusedPrefabsOnShutdown
```

If true, all prefabs will be unloaded on shutdown.

6.244 NetworkProjectConfig Class Reference

The core [Fusion](#) config file that is shared with all peers at startup.

Inherits [IConfigurationSanityCheck](#).

Public Types

- enum class [PeerModes](#)

Options for running one or multiple peers in one Unity instance. Multiple is useful for testing multiple players/clients inside of the Unity editor without needing to build executables. Each peer is assigned its own independent physics scene and [NetworkRunner](#) instance.

- enum class [ReplicationFeatures](#)

Public Member Functions

- int? [GetExecutionOrder](#) (Type type)
Get the execution order for a given type. If the type is registered, returns null.
- void [SanityCheck](#) ()
- override string [ToString](#) ()
ToString() implementation.

Static Public Member Functions

- static [NetworkProjectConfig Deserialize](#) (string data)
De-serialize a [NetworkProjectConfig](#) from a JSON string (typically sent by the Room's Creator).
- static string [Serialize](#) ([NetworkProjectConfig](#) config)
Serialize a [NetworkProjectConfig](#) into a JSON string.
- static void [UnloadGlobal](#) ()
Unloads [Global](#), if already loaded. If loading [Global](#) has faulted, resets the state and next call to the [Global](#) accessor will attempt to load the config again.

Public Attributes

- bool [AllowClientServerModesInWebGL](#) = false
Allow the use of Client-Server modes in WebGL builds. This is not recommended, as it can lead to degraded performance. Client-Server modes are not supported in WebGL builds by default, this setting allows its usage despite the limitations. Read more about the limitations of WebGL builds in the [Fusion](#) documentation at <https://doc.photonengine.com/fusion/v2/fusion-intro>.
- string[] [AssembliesToWeave](#)
Names of assemblies [Fusion](#) is going to weave. Not case sensitive.
- bool [CheckNetworkedPropertiesBeingEmpty](#) = false
If set, the weaver will check if [NetworkedAttribute](#) properties getters and setters are empty.
- bool [CheckRpcAttributeUsage](#) = false
If set, the weaver will check if [RpcAttribute](#) is used in types that do not support it. This requires all types to be scanned and can increase weaving duration.
- [EncryptionConfig EncryptionConfig](#) = new [EncryptionConfig](#)()
Reference to [EncryptionConfig](#) settings for this [NetworkProjectConfig](#)
- bool [EnqueueIncompleteSynchronousSpawns](#)
This flag changes the behaviour of [NetworkRunner.Spawn<T>](#) to return null (instead of throwing an exception) and [NetworkRunner.TrySpawn<T>](#) to return [NetworkSpawnStatus.Queued](#) if [Fusion](#) was unable to load a prefab synchronously (e.g. because it was [Addressable](#)). [Fusion](#) will enqueue the spawn and attempt to perform it the next frame, until successful. Useful for transition from [Fusion](#) 1.x.
- [HeapConfiguration Heap](#) = new [HeapConfiguration](#)()
Heap Settings
- bool [HideNetworkObjectInactivityGuard](#) = false
Inactive [NetworkObject](#) need special handling in case they get destroyed without ever being activated. This is achieved with adding a nested [GameObject](#) called "NetworkObjectInactivityGuard" that tracks the [OnDestroy](#) message. [HideNetworkObjectInactivityGuard](#) can be used to control whether these guards are visible in the hierarchy or not.
- [HostMigrationConfig HostMigration](#) = new [HostMigrationConfig](#)()
Reference to [HostMigration](#) settings for this [NetworkProjectConfig](#)
- bool [InvokeRenderInBatchMode](#) = true
Signal if the [SimulationBehaviour.Render](#) callbacks should be invoked in Batch Mode.
- [LagCompensationSettings LagCompensation](#) = new [LagCompensationSettings](#)()
Advanced lag compensation buffer settings.

- `NetworkConfiguration Network = new NetworkConfiguration()`
Reference to `NetworkConfiguration` settings for this `NetworkProjectConfig`.
- `NetworkSimulationConfiguration NetworkConditions = new NetworkSimulationConfiguration()`
Settings for simulating network conditions of latency and loss.
- `bool NetworkIdsObjectName`
Signal if the `NetworkId` of the `NetworkObject` should be included on the name of the `GameObject`
- `bool NullChecksForNetworkedProperties = true`
If set, the weaver will add a check to all [Networked] properties on each `NetworkBehaviour` to verify if owing `NetworkObject` has been attached to.
- `PeerModes PeerMode`
Setting for whether multiple peers can run per Unity instance (typically to allow easy testing of multiple peers inside of the editor).
- `NetworkPrefabTable PrefabTable = new NetworkPrefabTable()`
Reference to the `NetworkPrefabTable` instance for this `NetworkProjectConfig`.
- `SimulationConfig Simulation = new SimulationConfig()`
Reference to `SimulationConfig` settings for this `NetworkProjectConfig`.
- `TimeSyncConfiguration TimeSynchronizationOverride`
this can be used to override the time synchronization from code
- `string TypeId = CurrentTypeId`
Current `NetworkProjectConfig` Type ID
- `bool UseSerializableDictionary = true`
Use `Fusion.SerializableDictionary` to store [Networked] dictionary properties initial value. If unchecked, the weaver will emit `System.Generic.Dictionary` instead - a type that's not Unity-serializable, but custom serializers (e.g. Odin) may support it.
- `int Version = CurrentVersion`
Current `NetworkProjectConfig` version

Static Public Attributes

- static `NetworkRunner. BuildTypes`
Get the version information for the `Fusion.Runtime.dll`.
- `const string CurrentTypeId = nameof(NetworkProjectConfig)`
Current `NetworkProjectConfig` Type ID
- `const int CurrentVersion = 1`
Current `NetworkProjectConfig` version
- `const string DefaultResourceName = nameof(NetworkProjectConfig)`
Default file name for the `NetworkProjectConfig` asset

Properties

- static `NetworkProjectConfig Global` [get]
Reference for the default `NetworkProjectConfig`. By default, loads a resource named "NetworkProjectConfig". This behaviour can be changed with an attribute `FusionGlobalScriptableObjectLoaderMethodAttribute`.

6.244.1 Detailed Description

The core `Fusion` config file that is shared with all peers at startup.

6.244.2 Member Enumeration Documentation

6.244.2.1 PeerModes

```
enum PeerModes [strong]
```

Options for running one or multiple peers in one Unity instance. Multiple is useful for testing multiple players/clients inside of the Unity editor without needing to build executables. Each peer is assigned its own independent physics scene and [NetworkRunner](#) instance.

Enumerator

Single	This is the normal use case, where every build and the editor run a single server, host or client peer.
Multiple	This is the optional use case, which allows running multiple peers in the Unity editor, or in a build.

6.244.2.2 ReplicationFeatures

```
enum ReplicationFeatures [strong]
```

Eventual Consistency [NetworkObject](#) state replication options.

Scheduling enables automatic prioritization of objects when culling occurs (when Object's are not replicated due to exceeding per tick data limits, they increase in priority on the following [Tick](#)).

Interest Management enables [NetworkObject](#) Area Of Interest and Explicit Interest features.

Enumerator

None	No special replication handling. This setting is ideal if your project never exceeds per tick data limits during gameplay.
Scheduling	When changed Network Objects are not replicated by the server to a client due to culling (data per tick limit was reached) the server increases the priority of that Network Object for the next outgoing Tick update to that client.
SchedulingAndInterestManagement	In addition to scheduling, Interest Management features are also enabled (Area Of Interest and Explicit Interest).

6.244.3 Member Function Documentation

6.244.3.1 Deserialize()

```
static NetworkProjectConfig Deserialize (
    string data ) [static]
```

De-serialize a [NetworkProjectConfig](#) from a JSON string (typically sent by the Room's Creator).

Parameters

<i>data</i>	JSON string of a serialized NetworkProjectConfig
-------------	--

Returns

[NetworkProjectConfig](#) reference de-serialized from JSON string

6.244.3.2 GetExecutionOrder()

```
int? GetExecutionOrder (
    Type type )
```

Get the execution order for a given type. If the type is registered, returns null.

Parameters

<i>type</i>	Type to check for the execution order.
-------------	--

Returns

Execution order for the type, or null if not registered.

6.244.3.3 Serialize()

```
static string Serialize (
    NetworkProjectConfig config ) [static]
```

Serialize a [NetworkProjectConfig](#) into a JSON string.

Parameters

<i>config</i>	NetworkProjectConfig reference
---------------	--

Returns

JSON String

6.244.3.4 ToString()

```
override string ToString ( )
```

[ToString\(\)](#) implementation.

6.244.3.5 UnloadGlobal()

```
static void UnloadGlobal ( ) [static]
```

Unloads [Global](#), if already loaded. If loading [Global](#) has faulted, resets the state and next call to the [Global](#) accessor will attempt to load the config again.

6.244.4 Member Data Documentation

6.244.4.1 AllowClientServerModesInWebGL

```
bool AllowClientServerModesInWebGL = false
```

Allow the use of Client-Server modes in WebGL builds. This is not recommended, as it can lead to degraded performance. Client-Server modes are not supported in WebGL builds by default, this setting allows its usage despite the limitations. Read more about the limitations of WebGL builds in the [Fusion](https://doc.photonengine.com/fusion/v2/fusion-intro) documentation at <https://doc.photonengine.com/fusion/v2/fusion-intro>.

6.244.4.2 AssembliesToWeave

```
string [ ] AssembliesToWeave
```

Initial value:

```
= new string[] {  
    "Fusion.Unity",  
    "Assembly-CSharp",  
    "Assembly-CSharp-firstpass",  
    "Fusion.Addons.Physics",  
    "Fusion.Addons.FSM",  
}
```

Names of assemblies [Fusion](#) is going to weave. Not case sensitive.

6.244.4.3 BuildTypes

```
NetworkRunner. BuildTypes [static]
```

Get the version information for the Fusion.Runtime.dll.

6.244.4.4 CheckNetworkedPropertiesBeingEmpty

```
bool CheckNetworkedPropertiesBeingEmpty = false
```

If set, the weaver will check if [NetworkedAttribute](#) properties getters and setters are empty.

6.244.4.5 CheckRpcAttributeUsage

```
bool CheckRpcAttributeUsage = false
```

If set, the weaver will check if [RpcAttribute](#) is used in types that do not support it. This requires all types to be scanned and can increase weaving duration.

6.244.4.6 CurrentTypeId

```
const string CurrentTypeId = nameof(NetworkProjectConfig) [static]
```

Current [NetworkProjectConfig](#) Type ID

6.244.4.7 CurrentVersion

```
const int CurrentVersion = 1 [static]
```

Current [NetworkProjectConfig](#) version

6.244.4.8 DefaultResourceName

```
const string DefaultResourceName = nameof(NetworkProjectConfig) [static]
```

Default file name for the [NetworkProjectConfig](#) asset

6.244.4.9 EncryptionConfig

```
EncryptionConfig EncryptionConfig = new EncryptionConfig()
```

Reference to [EncryptionConfig](#) settings for this [NetworkProjectConfig](#)

6.244.4.10 EnqueueIncompleteSynchronousSpawns

```
bool EnqueueIncompleteSynchronousSpawns
```

This flag changes the behaviour of [NetworkRunner.Spawn<T>](#) to return null (instead of throwing an exception) and [NetworkRunner.TrySpawn<T>](#) to return [NetworkSpawnStatus.Queued](#) if [Fusion](#) was unable to load a prefab synchronously (e.g. because it was [Addressable](#)). [Fusion](#) will enqueue the spawn and attempt to perform it the next frame, until successful. Useful for transition from [Fusion 1.x](#).

6.244.4.11 Heap

```
HeapConfiguration Heap = new HeapConfiguration()
```

Heap Settings

6.244.4.12 HideNetworkObjectInactivityGuard

```
bool HideNetworkObjectInactivityGuard = false
```

Inactive [NetworkObject](#) need special handling in case they get destroyed without ever being activated. This is achieved with adding a nested [GameObject](#) called "NetworkObjectInactivityGuard" that tracks the [OnDestroy](#) message. [HideNetworkObjectInactivityGuard](#) can be used to control whether these guards are visible in the hierarchy or not.

6.244.4.13 HostMigration

```
HostMigrationConfig HostMigration = new HostMigrationConfig()
```

Reference to [HostMigration](#) settings for this [NetworkProjectConfig](#)

6.244.4.14 InvokeRenderInBatchMode

```
bool InvokeRenderInBatchMode = true
```

Signal if the [SimulationBehaviour.Render](#) callbacks should be invoked in Batch Mode.

6.244.4.15 LagCompensation

```
LagCompensationSettings LagCompensation = new LagCompensationSettings()
```

Advanced lag compensation buffer settings.

6.244.4.16 Network

```
NetworkConfiguration Network = new NetworkConfiguration()
```

Reference to [NetworkConfiguration](#) settings for this [NetworkProjectConfig](#).

6.244.4.17 NetworkConditions

```
NetworkSimulationConfiguration NetworkConditions = new NetworkSimulationConfiguration()
```

Settings for simulating network conditions of latency and loss.

6.244.4.18 NetworkIdIsObjectName

```
bool NetworkIdIsObjectName
```

Signal if the [NetworkId](#) of the [NetworkObject](#) should be included on the name of the [GameObject](#)

6.244.4.19 NullChecksForNetworkedProperties

```
bool NullChecksForNetworkedProperties = true
```

If set, the weaver will add a check to all [Networked] properties on each [NetworkBehaviour](#) to verify if owing [NetworkObject](#) has been attached to.

6.244.4.20 PeerMode

```
PeerModes PeerMode
```

Setting for whether multiple peers can run per Unity instance (typically to allow easy testing of multiple peers inside of the editor).

6.244.4.21 PrefabTable

```
NetworkPrefabTable PrefabTable = new NetworkPrefabTable()
```

Reference to the [NetworkPrefabTable](#) instance for this [NetworkProjectConfig](#).

6.244.4.22 Simulation

```
SimulationConfig Simulation = new SimulationConfig()
```

Reference to [SimulationConfig](#) settings for this [NetworkProjectConfig](#).

6.244.4.23 TimeSynchronizationOverride

```
TimeSyncConfiguration TimeSynchronizationOverride
```

this can be used to override the time synchronization from code

6.244.4.24 TypeId

```
string TypeId = CurrentTypeId
```

Current [NetworkProjectConfig](#) Type ID

6.244.4.25 UseSerializableDictionary

```
bool UseSerializableDictionary = true
```

Use [Fusion.SerializableDictionary](#) to store [Networked] dictionary properties initial value. If unchecked, the weaver will emit `System.Generic.Dictionary` instead - a type that's not Unity-serializable, but custom serializers (e.g. Odin) may support it.

6.244.4.26 Version

```
int Version = CurrentVersion
```

Current [NetworkProjectConfig](#) version

6.244.5 Property Documentation

6.244.5.1 Global

`NetworkProjectConfig` Global [static], [get]

Reference for the default `NetworkProjectConfig`. By default, loads a resource named "NetworkProjectConfig". This behaviour can be changed with an attribute `FusionGlobalScriptableObjectLoaderMethodAttribute`.

6.245 NetworkProjectConfigAsset Class Reference

Manages and references the current instance of `NetworkProjectConfig`

Inherits `FusionGlobalScriptableObject`< `NetworkProjectConfigAsset` >.

Classes

- struct `SerializableSimulationBehaviourMeta`

An auto-generated list containing meta information about all the `SimulationBehaviours` in the project, e.g. execution order.

Static Public Member Functions

- static bool `TryGetGlobal` (out `NetworkProjectConfigAsset` global)

Try to get the current `NetworkProjectConfig` instance.

- static void `UnloadGlobal` ()

Unload the current `NetworkProjectConfig` instance.

Public Attributes

- `SerializableSimulationBehaviourMeta[]` `BehaviourMeta` = `Array.Empty`<`SerializableSimulationBehaviourMeta`>()

An auto-generated list containing meta information about all the `SimulationBehaviours` in the project, e.g. execution order.

- `NetworkProjectConfig` `Config` = `new` `NetworkProjectConfig`()

The current `NetworkProjectConfig` instance.

- `NetworkPrefabTableOptions` `PrefabOptions` = `NetworkPrefabTableOptions.Default`

Options for the `NetworkPrefabTable`.

- `List`< `INetworkPrefabSource` > `Prefabs` = `new` `List`<`INetworkPrefabSource`>()

An auto-generated list containing source information (e.g. Resource path, address, static reference) for all the prefabs that can be spawned, i.e. the ones with `NetworkObject` component and `NetworkObject.IsSpawnable` enabled. Additional prefabs can registered at runtime with `NetworkPrefabTable.TryAddSource`.

Protected Member Functions

- override void `OnDisable` ()

Unloads all prefabs.

Properties

- static [NetworkProjectConfigAsset Global](#) [get]
The current [NetworkProjectConfig](#) instance.
- static bool [IsGlobalLoaded](#) [get]
True if the [NetworkProjectConfig](#) instance exists, otherwise false.

Additional Inherited Members

6.245.1 Detailed Description

Manages and references the current instance of [NetworkProjectConfig](#)

6.245.2 Member Function Documentation

6.245.2.1 OnDisable()

```
override void OnDisable ( ) [protected], [virtual]
```

Unloads all prefabs.

Reimplemented from [FusionGlobalScriptableObject< NetworkProjectConfigAsset >](#).

6.245.2.2 TryGetGlobal()

```
static bool TryGetGlobal (
    out NetworkProjectConfigAsset global ) [static]
```

Try to get the current [NetworkProjectConfig](#) instance.

Parameters

<i>global</i>	NetworkProjectConfig instance if it exists, otherwise null.
---------------	---

Returns

True if the [NetworkProjectConfig](#) instance exists, otherwise false.

6.245.2.3 UnloadGlobal()

```
static void UnloadGlobal ( ) [static]
```

Unload the current [NetworkProjectConfig](#) instance.

6.245.3 Member Data Documentation

6.245.3.1 BehaviourMeta

```
SerializableSimulationBehaviourMeta [ ] BehaviourMeta = Array.Empty<SerializableSimulationBehaviourMeta>()
```

An auto-generated list containing meta information about all the [SimulationBehaviours](#) in the project, e.g. execution order.

6.245.3.2 Config

```
NetworkProjectConfig Config = new NetworkProjectConfig()
```

The current [NetworkProjectConfig](#) instance.

6.245.3.3 PrefabOptions

```
NetworkPrefabTableOptions PrefabOptions = NetworkPrefabTableOptions.Default
```

Options for the [NetworkPrefabTable](#).

6.245.3.4 Prefabs

```
List<INetworkPrefabSource> Prefabs = new List<INetworkPrefabSource>()
```

An auto-generated list containing source information (e.g. Resource path, address, static reference) for all the prefabs that can be spawned, i.e. the ones with [NetworkObject](#) component and [NetworkObject.IsSpawnable](#) enabled. Additional prefabs can be registered at runtime with [NetworkPrefabTable.TryAddSource](#).

6.245.4 Property Documentation

6.245.4.1 Global

```
NetworkProjectConfigAsset Global [static], [get]
```

The current [NetworkProjectConfig](#) instance.

6.245.4.2 IsGlobalLoaded

```
bool IsGlobalLoaded [static], [get]
```

True if the [NetworkProjectConfig](#) instance exists, otherwise false.

6.246 NetworkProjectConfigAsset.SerializableSimulationBehaviourMeta Struct Reference

An auto-generated list containing meta information about all the [SimulationBehaviours](#) in the project, e.g. execution order.

Public Attributes

- int [ExecutionOrder](#)
The execution order of the [SimulationBehaviour](#).
- [SerializableType](#)< [SimulationBehaviour](#) > [Type](#)
The type of the [SimulationBehaviour](#).

6.246.1 Detailed Description

An auto-generated list containing meta information about all the [SimulationBehaviours](#) in the project, e.g. execution order.

6.246.2 Member Data Documentation

6.246.2.1 ExecutionOrder

```
int ExecutionOrder
```

The execution order of the [SimulationBehaviour](#).

6.246.2.2 Type

`SerializableType<SimulationBehaviour>` Type

The type of the `SimulationBehaviour`.

6.247 NetworkRNG Struct Reference

PCG32 random generator, 16 bytes in size. <http://www.pcg-random.org>

Inherits `INetworkStruct`.

Public Member Functions

- `NetworkRNG` (Int32 seed)
 - Creates a new instance of `NetworkRNG` with a random seed.*
- double `Next` ()
 - Generates a random double value within the inclusive range [0, 1].*
- double `NextExclusive` ()
 - Generates a random double value within the exclusive range [0, 1).*
- int `NextInt32` ()
 - Generates a random integer value within the range of `int.MinValue` to `int.MaxValue`.*
- float `NextSingle` ()
 - Generates a random float value within the inclusive range [0, 1].*
- float `NextSingleExclusive` ()
 - Generates a random float value within the exclusive range [0, 1).*
- uint `NextUInt32` ()
 - Generates a random unsigned integer value within the range of 0 to `uint.MaxValue`.*
- Int32 `RangeExclusive` (Int32 minInclusive, Int32 maxExclusive)
 - Returns a random Int32 within [minInclusive, maxExclusive) (range is exclusive). If minInclusive and maxExclusive are equal, then the "exclusive rule" is ignored and minInclusive will be returned. If minInclusive is greater than maxExclusive, then the numbers are automatically swapped.*
- UInt32 `RangeExclusive` (UInt32 minInclusive, UInt32 maxExclusive)
 - Returns a random UInt32 within [minInclusive, maxExclusive) (range is exclusive). If minInclusive and maxExclusive are equal, then the "exclusive rule" is ignored and minInclusive will be returned. If minInclusive is greater than maxExclusive, then the numbers are automatically swapped.*
- Double `RangeInclusive` (Double minInclusive, Double maxInclusive)
 - Returns a random Double within [minInclusive, maxInclusive] (range is inclusive). If minInclusive is greater than maxInclusive, then the numbers are automatically swapped.*
- Int32 `RangeInclusive` (Int32 minInclusive, Int32 maxInclusive)
 - Returns a random Int32 within [minInclusive, maxInclusive] (range is inclusive). If minInclusive is greater than maxInclusive, then the numbers are automatically swapped.*
- Single `RangeInclusive` (Single minInclusive, Single maxInclusive)
 - Returns a random Single within [minInclusive, maxInclusive] (range is inclusive). If minInclusive is greater than maxInclusive, then the numbers are automatically swapped.*
- UInt32 `RangeInclusive` (UInt32 minInclusive, UInt32 maxInclusive)
 - Returns a random UInt32 within [minInclusive, maxInclusive] (range is inclusive). If minInclusive is greater than maxInclusive, then the numbers are automatically swapped.*
- override string `ToString` ()
 - String representation of the RNG state.*

Static Public Attributes

- const UInt32 `MAX` = UInt32.MaxValue
Maximum allowed value
- const int `SIZE` = 16
Size of the struct in bytes.

Properties

- `NetworkRNG Peek` [get]
Returns the same RNG instance.

6.247.1 Detailed Description

PCG32 random generator, 16 bytes in size. <http://www.pcg-random.org>

6.247.2 Constructor & Destructor Documentation

6.247.2.1 NetworkRNG()

```
NetworkRNG (
    Int32 seed )
```

Creates a new instance of `NetworkRNG` with a random seed.

Parameters

<code>seed</code>	Seed value.
-------------------	-------------

6.247.3 Member Function Documentation

6.247.3.1 Next()

```
double Next ( )
```

Generates a random double value within the inclusive range [0, 1].

Returns

A random double value between 0 and 1, inclusive.

6.247.3.2 NextExclusive()

```
double NextExclusive ( )
```

Generates a random double value within the exclusive range [0, 1).

Returns

A random double value between 0 (inclusive) and 1 (exclusive).

6.247.3.3 NextInt32()

```
int NextInt32 ( )
```

Generates a random integer value within the range of `int.MinValue` to `int.MaxValue`.

Returns

A random integer value between `int.MinValue` and `int.MaxValue`, inclusive.

6.247.3.4 NextSingle()

```
float NextSingle ( )
```

Generates a random float value within the inclusive range [0, 1].

Returns

A random float value between 0 and 1, inclusive.

6.247.3.5 NextSingleExclusive()

```
float NextSingleExclusive ( )
```

Generates a random float value within the exclusive range [0, 1).

Returns

A random float value between 0 (inclusive) and 1 (exclusive).

6.247.3.6 NextUInt32()

```
uint NextUInt32 ( )
```

Generates a random unsigned integer value within the range of 0 to `uint.MaxValue`.

Returns

A random unsigned integer value between 0 and `uint.MaxValue`, inclusive.

6.247.3.7 RangeExclusive() [1/2]

```
Int32 RangeExclusive (
    Int32 minInclusive,
    Int32 maxExclusive )
```

Returns a random `Int32` within `[minInclusive, maxExclusive)` (range is exclusive). If `minInclusive` and `maxExclusive` are equal, then the "exclusive rule" is ignored and `minInclusive` will be returned. If `minInclusive` is greater than `maxExclusive`, then the numbers are automatically swapped.

6.247.3.8 RangeExclusive() [2/2]

```
UInt32 RangeExclusive (
    UInt32 minInclusive,
    UInt32 maxExclusive )
```

Returns a random `UInt32` within `[minInclusive, maxExclusive)` (range is exclusive). If `minInclusive` and `maxExclusive` are equal, then the "exclusive rule" is ignored and `minInclusive` will be returned. If `minInclusive` is greater than `maxExclusive`, then the numbers are automatically swapped.

6.247.3.9 RangeInclusive() [1/4]

```
Double RangeInclusive (
    Double minInclusive,
    Double maxInclusive )
```

Returns a random `Double` within `[minInclusive, maxInclusive]` (range is inclusive). If `minInclusive` is greater than `maxInclusive`, then the numbers are automatically swapped.

6.247.3.10 RangeInclusive() [2/4]

```
Int32 RangeInclusive (
    Int32 minInclusive,
    Int32 maxInclusive )
```

Returns a random Int32 within [minInclusive, maxInclusive] (range is inclusive). If minInclusive is greater than maxInclusive, then the numbers are automatically swapped.

6.247.3.11 RangeInclusive() [3/4]

```
Single RangeInclusive (
    Single minInclusive,
    Single maxInclusive )
```

Returns a random Single within [minInclusive, maxInclusive] (range is inclusive). If minInclusive is greater than maxInclusive, then the numbers are automatically swapped.

6.247.3.12 RangeInclusive() [4/4]

```
UInt32 RangeInclusive (
    UInt32 minInclusive,
    UInt32 maxInclusive )
```

Returns a random UInt32 within [minInclusive, maxInclusive] (range is inclusive). If minInclusive is greater than maxInclusive, then the numbers are automatically swapped.

6.247.3.13 ToString()

```
override string ToString ( )
```

String representation of the RNG state.

6.247.4 Member Data Documentation

6.247.4.1 MAX

```
const UInt32 MAX = UInt32.MaxValue [static]
```

Maximum allowed value

6.247.4.2 SIZE

```
const int SIZE = 16 [static]
```

Size of the struct in bytes.

6.247.5 Property Documentation

6.247.5.1 Peek

```
NetworkRNG Peek [get]
```

Returns the same RNG instance.

6.248 NetworkRpcStaticWeavedInvokerAttribute Class Reference

Network Rpc Static Weaved Invoker Attribute Contains info about a static weaved RPC Method

Inherits Attribute.

Public Member Functions

- [NetworkRpcStaticWeavedInvokerAttribute](#) (string key)
NetworkRpcStaticWeavedInvokerAttribute Constructor

Properties

- string [Key](#) [get]
RPC Key

6.248.1 Detailed Description

Network Rpc Static Weaved Invoker Attribute Contains info about a static weaved RPC Method

6.248.2 Constructor & Destructor Documentation

6.248.2.1 NetworkRpcStaticWeavedInvokerAttribute()

```
NetworkRpcStaticWeavedInvokerAttribute (  
    string key )
```

[NetworkRpcStaticWeavedInvokerAttribute](#) Constructor

Parameters

<i>key</i>	Key
------------	---------------------

6.248.3 Property Documentation

6.248.3.1 Key

`string Key [get]`

RPC Key

6.249 NetworkRpcWeavedInvokerAttribute Class Reference

Network Rpc Weaved Invoker Attribute Contains info about a weaved RPC Method

Inherits Attribute.

Public Member Functions

- [NetworkRpcWeavedInvokerAttribute](#) (int key, int sources, int targets)
NetworkRpcWeavedInvokerAttribute Constructor

Properties

- int [Key](#) [get]
RPC Key
- int [Sources](#) [get]
RPC Sources
- int [Targets](#) [get]
RPC Targets

6.249.1 Detailed Description

Network Rpc Weaved Invoker Attribute Contains info about a weaved RPC Method

6.249.2 Constructor & Destructor Documentation

6.249.2.1 NetworkRpcWeavedInvokerAttribute()

```
NetworkRpcWeavedInvokerAttribute (
    int key,
    int sources,
    int targets )
```

[NetworkRpcWeavedInvokerAttribute](#) Constructor

Parameters

<i>key</i>	Key
<i>sources</i>	Sources
<i>targets</i>	Targets

6.249.3 Property Documentation

6.249.3.1 Key

```
int Key [get]
```

RPC Key

6.249.3.2 Sources

```
int Sources [get]
```

RPC Sources

6.249.3.3 Targets

```
int Targets [get]
```

RPC Targets

6.250 NetworkRunner Class Reference

Host Migration related code in order to get a copy of the [Simulation](#) State

Inherits [Behaviour](#), and [Simulation.ICallbacks](#).

Public Types

- enum class [BuildTypes](#)
Enumeration of Fusion.Runtime.dll options.
- enum class [States](#)
Initialization stages of Fusion

Public Member Functions

- void [AddCallbacks](#) (params [INetworkRunnerCallbacks](#)[] callbacks)

Register an [INetworkRunnerCallbacks](#) instance for callbacks from this [NetworkRunner](#).
- void [AddGlobal](#) ([SimulationBehaviour](#) instance)

Add and register a [SimulationBehaviour](#) to this [NetworkRunner](#). Note: It should NOT be a [NetworkBehaviour](#)
- void [AddPlayerAreaOfInterest](#) ([PlayerRef](#) player, [Vector3](#) center, float radius)

Call this every [FixedUpdateNetwork](#) to add an area of interest for a player. Should only be called from the Host/Server in Server client mode. Should only be called for the local player in shared mode.
- void [Attach](#) ([NetworkObject](#) networkObject, [PlayerRef?](#) inputAuthority=null, bool allocate=true, bool? masterClientObjectOverride=null)

Attaches a user-created network object to the network.
- void [Attach](#) ([NetworkObject](#)[] networkObjects, [PlayerRef?](#) inputAuthority=null, bool allocate=true, bool? masterClientObjectOverride=null)

Attach and assign to this [NetworkRunner](#) the [NetworkObject](#) provided. Used internally from the default implementation of [INetworkSceneManager](#) to register scene objects.
- void [ClearPlayerAreaOfInterest](#) ([PlayerRef](#) player)

Clears the area of interest for a player. This can only be called from the server/host
- delegate void [CloudConnectionLostHandler](#) ([NetworkRunner](#) networkRunner, [ShutdownReason](#) shutdownReason, bool reconnecting)

If this callback is implemented, the default behavior of disconnecting when the cloud connection is lost in server/client or host/client mode is disabled. This callback will be invoked instead, leaving the decision up to the user. It is called both on the server/host and on the client.
- void [Despawn](#) ([NetworkObject](#) networkObject)

Destroys a [NetworkObject](#).
- void [DestroySingleton](#)< T > ()

Removes a specific [SimulationBehaviour](#) from this [NetworkRunner](#) gameobject, if it exists.
- void [Disconnect](#) ([PlayerRef](#) player, byte[] token=null)

Disconnects a player from the server.
- bool [EnsureRunnerScenelsActive](#) (out Scene previousActiveScene)

Ensures the scene of this runner is active and returns the previous active scene
- bool [Exists](#) ([NetworkId](#) id)

Returns if the [Fusion.Simulation](#) contains a [NetworkObject](#) with given id in the current State.
- bool [Exists](#) ([NetworkObject](#) obj)

Returns if the [Fusion.Simulation](#) contains a reference to a [NetworkObject](#) in the current State.
- [NetworkObject](#) [FindObject](#) ([NetworkId](#) networkId)

Get the [NetworkObject](#) instance for this [NetworkRunner](#) from a [NetworkId](#).
- [SimulationBehaviour](#)[] [GetAllBehaviours](#) (Type type)

Returns an array of all [SimulationBehaviour](#) instances registered with this [NetworkRunner](#).
- List< T > [GetAllBehaviours](#)< T > ()

Get a list with all behaviours of the desired type that are registered on the [NetworkRunner](#).
- void [GetAllBehaviours](#)< T > (List< T > result)

Add on the list all behaviours of the desired type that are registered on the [NetworkRunner](#). Note: The list will not be cleared before adding the results.
- List< [NetworkObject](#) > [GetAllNetworkObjects](#) ()

Retrieves a list of all network objects in the simulation.
- void [GetAllNetworkObjects](#) (List< [NetworkObject](#) > result)

Populate a list with all network objects in the simulation.
- void [GetAreaOfInterestGizmoData](#) (List<(Vector3 center, Vector3 size, int playerCount, int objectCount)> result)

Populates the provided list with data about the current Area of Interest (AOI) cells. Each element in the list represents one AOI cell.
- T? [GetInputForPlayer](#)< T > ([PlayerRef](#) player)

- Returns the [NetworkInput](#) data from *player*, converted to the indicated [INetworkInput](#).
- [SimulationBehaviourListScope GetInterfaceListHead](#) (Type type, int index, out [SimulationBehaviour](#) head)
Get the interface list head.
 - [SimulationBehaviour GetInterfaceListNext](#) ([SimulationBehaviour](#) behaviour)
Get the next behaviour
 - [SimulationBehaviour GetInterfaceListPrev](#) ([SimulationBehaviour](#) behaviour)
Get the previous behaviour
 - int [GetInterfaceListsCount](#) (Type type)
Get the number of interfaces of the desired type that are registered on the behaviour updater.
 - void [GetMemorySnapshot](#) ([MemoryStatisticsSnapshot.TargetAllocator](#) targetAllocator, ref [MemoryStatisticsSnapshot](#) snapshot)
Gets a memory snapshot of the simulation.
 - List< [NetworkId](#) > [GetObjectsInAreaOfInterestForPlayer](#) ([PlayerRef](#) player)
Retrieves a list of network object IDs that are in the area of interest for the specified player. Server only.
 - [PhysicsScene GetPhysicsScene](#) ()
Get the 3D Physics scene being used by this Runner.
 - [PhysicsScene2D GetPhysicsScene2D](#) ()
Get the 2D Physics scene being used by this Runner.
 - int? [GetPlayerActorId](#) ([PlayerRef](#) player)
Gets Player's Actor Number (ID).
 - byte[] [GetPlayerConnectionToken](#) ([PlayerRef](#) player=default)
Returns a copy of the Connection Token used by a Player when connecting to this Server. Only available on Server. It will return null if running on a Client or the Connection token is missing
 - [ConnectionType GetPlayerConnectionType](#) ([PlayerRef](#) player)
Return the [ConnectionType](#) with a Remote [PlayerRef](#). Valid only when invoked from a Server ([NetworkRunner.IsServer](#))
 - [NetworkObject GetPlayerObject](#) ([PlayerRef](#) player)
Gets the network object associated with a specific player
 - double [GetPlayerRtt](#) ([PlayerRef](#) playerRef)
Returns the player round trip time (ping) in seconds
 - string [GetPlayerUserId](#) ([PlayerRef](#) player=default)
Gets Player's UserID.
 - [NetworkInput? GetRawInputForPlayer](#) ([PlayerRef](#) player)
Returns the unconverted unsafe [NetworkInput](#) for the indicated player.
 - IEnumerable< [NetworkObject](#) > [GetResumeSnapshotNetworkObjects](#) ()
Iterate over the old [NetworkObjects](#) from the Resume Snapshot
 - IEnumerable<([NetworkObject](#), [NetworkObjectHeaderPtr](#))> [GetResumeSnapshotNetworkSceneObjects](#) ()
Iterate over the Scene [NetworkObjects](#) from the Resume Snapshot while giving the reference of the old Snapshot data associated with that particular Scene Object
 - [RpcTargetStatus GetRpcTargetStatus](#) ([PlayerRef](#) target)
Return the [RpcTargetStatus](#) for a specific player.
 - [SceneRef GetSceneRef](#) ([GameObject](#) gameObj)
 - [SceneRef GetSceneRef](#) (string sceneNameOrPath)
 - T [GetSingleton](#)< T > ()
Ensures that a specific [SimulationBehaviour](#) component exists on this [NetworkRunner](#) gameobject.
 - bool [HasAnyActiveConnections](#) ()
 - bool [HasSingleton](#)< T > ()
Returns if a given [SimulationBehaviour](#) is present in this [NetworkRunner](#) gameobject.
 - [GameObject InstantiateInRunnerScene](#) ([GameObject](#) original)
Instantiates an object in the scene of this runner
 - [GameObject InstantiateInRunnerScene](#) ([GameObject](#) original, [Vector3](#) position, [Quaternion](#) rotation)
Instantiates an object in the scene of this runner

- T [InstantiateInRunnerScene< T >](#) (T original)
 - Instantiates an object in the scene of this runner*
- T [InstantiateInRunnerScene< T >](#) (T original, Vector3 position, Quaternion rotation)
 - Instantiates an object in the scene of this runner*
- void [InvokeSceneLoadDone](#) (in [SceneLoadDoneArgs](#) info)
 - Invoke [INetworkRunnerCallbacks.OnSceneLoadDone\(NetworkRunner\)](#) on all implementations*
- void [InvokeSceneLoadStart](#) ([SceneRef](#) sceneRef)
 - Invoke [INetworkRunnerCallbacks.OnSceneLoadStart\(NetworkRunner\)](#) on all implementations*
- bool? [IsInterestedIn](#) ([NetworkObject](#) obj, [PlayerRef](#) player)
 - Test if a player has Interest in a [NetworkObject](#).*
- bool [IsValidPlayer](#) ([PlayerRef](#) player)
 - Checks if the provided player is valid in the current simulation.*
- async Task< [StartGameResult](#) > [JoinSessionLobby](#) ([SessionLobby](#) sessionLobby, string lobbyID=null, [AuthenticationValues](#) authentication=null, [FusionAppSettings](#) customAppSettings=null, bool? useDefault↔ CloudPorts=false, [CancellationToken](#) cancellationToken=default, bool useCachedRegions=true)
 - Join the Peer to a specific Lobby, either a prebuild or a custom one.*
- [NetworkSceneAsyncOp LoadScene](#) ([SceneRef](#) sceneRef, [LoadSceneMode](#) loadSceneMode=Load↔ [SceneMode](#).Single, [LocalPhysicsMode](#) localPhysicsMode=[LocalPhysicsMode](#).None, bool setActiveOn↔ Load=DefaultSetActiveOnLoad)
 - Loads a scene*
- [NetworkSceneAsyncOp LoadScene](#) ([SceneRef](#) sceneRef, [LoadSceneParameters](#) parameters, bool set↔ ActiveOnLoad=DefaultSetActiveOnLoad)
 - Loads a scene*
- [NetworkSceneAsyncOp LoadScene](#) (string sceneName, [LoadSceneMode](#) loadSceneMode=Load↔ [SceneMode](#).Single, [LocalPhysicsMode](#) localPhysicsMode=[LocalPhysicsMode](#).None, bool setActiveOn↔ Load=DefaultSetActiveOnLoad)
 - Loads a scene*
- [NetworkSceneAsyncOp LoadScene](#) (string sceneName, [LoadSceneParameters](#) parameters, bool set↔ ActiveOnLoad=DefaultSetActiveOnLoad)
 - Loads a scene*
- void [MakeDontDestroyOnLoad](#) ([GameObject](#) obj)
 - Mark an object as [DontDestroyOnLoad](#).*
- bool [MoveGameObjectToSameScene](#) ([GameObject](#) gameObj, [GameObject](#) other)
 - Moves a [GameObject](#) to the same scene as another [GameObject](#)*
- bool [MoveGameObjectToScene](#) ([GameObject](#) gameObj, [SceneRef](#) sceneRef)
 - Moves a [GameObject](#) to a specific scene*
- void [MoveToRunnerScene](#) ([GameObject](#) instance, [SceneRef](#)? targetSceneRef=null)
 - Moves an object to the scene of this runner*
- void [MoveToRunnerScene< T >](#) (T component)
 - Moves an object to the scene of this runner*
- delegate void [ObjectDelegate](#) ([NetworkRunner](#) runner, [NetworkObject](#) obj)
 - Delegate type for object callback*
- delegate void [OnBeforeSpawned](#) ([NetworkRunner](#) runner, [NetworkObject](#) obj)
 - Delegate type for on before spawned callback*
- async Task< bool > [PushHostMigrationSnapshot](#) ()
 - Compute and send a [Host Migration Snapshot](#) to the [Photon Cloud](#)*
- int [RegisterSceneObjects](#) ([SceneRef](#) scene, [NetworkObject](#)[] objects, [NetworkSceneLoadId](#) loadId=default)
 - Registers scene objects to the network.*
- void [ReleaseStateAuthority](#) ([NetworkId](#) id)
 - Releases state authority for a given [NetworkId](#) on shared mode.*
- void [RemoveCallbacks](#) (params [INetworkRunnerCallbacks](#)[] callbacks)
 - Unregister an [INetworkRunnerCallbacks](#) instance for callbacks from this [NetworkRunner](#).*
- void [RemoveGlobal](#) ([SimulationBehaviour](#) instance)

- Removes a specific [SimulationBehaviour](#) from this [NetworkObject](#) gameobject, if it exists.*

 - void [RenderInternal](#) ()
 - This method is meant to be called by [INetworkRunnerUpdater](#).*
 - void [RequestStateAuthority](#) ([NetworkId](#) id)
 - Requests state authority for a specified [NetworkId](#) on shared mode.*
 - void [SendReliableDataToPlayer](#) ([PlayerRef](#) player, [ReliableKey](#) key, byte[] data)
 - Sends a reliable data buffer to a target player.*
 - void [SendReliableDataToServer](#) ([ReliableKey](#) key, byte[] data)
 - Sends a reliable data buffer to the server.*
 - void [SendRpc](#) ([SimulationMessage](#) *message)
 - Sends RPC message. Not meant to be used directly, [ILWeaver](#) calls this.*
 - void [SendRpc](#) ([SimulationMessage](#) *message, out [RpcSendResult](#) info)
 - Sends RPC message. Not meant to be used directly, [ILWeaver](#) calls this.*
 - void [SetAreaOfInterestCellSize](#) (int size)
 - Set the area of interest cell size*
 - void [SetAreaOfInterestGrid](#) (int x, int y, int z)
 - Set the area of interest grid dimensions*
 - void [SetBehaviourReplicateTo](#) ([NetworkBehaviour](#) behaviour, [PlayerRef](#) player, bool replicate)
 - Controls if a specific network behaviours state is replicated to a player or not*
 - void [SetBehaviourReplicateToAll](#) ([NetworkBehaviour](#) behaviour, bool replicate)
 - Controls if a specific network behaviours state is replicated to all players or not*
 - bool [SetIsSimulated](#) ([NetworkObject](#) obj, bool simulate)
 - Sets the simulation state for this object, if it takes part in the [NetworkFixedUpdate](#), etc. In shared mode a client cannot change the simulation state of [NetworkObjects](#) it does not have state authority over.*
 - void [SetMasterClient](#) ([PlayerRef](#) player)
 - Promote a player to be the new master client. Only the master client is able to call this method*
 - void [SetPlayerAlwaysInterested](#) ([PlayerRef](#) player, [NetworkObject](#) networkObject, bool alwaysInterested)
 - Flags this player as always interested in this object. Means it does not have to be in a players area of interest to be replicated. Only the [NetworkObject](#) State Authority can set interest.*
 - void [SetPlayerObject](#) ([PlayerRef](#) player, [NetworkObject](#) networkObject)
 - Sets the network object associated with this player*
 - void [SetSimulateMultiPeerPhysics](#) (bool value)
 - Set the value for simulating physics scenes when using multi-peer. Restores the default physics simulation settings if false, overrides the default if true. (2D: [SimulationMode2D.Script](#) | 3D: [AutoSimulation](#) = false)*
 - Task [Shutdown](#) (bool destroyGameObject=true, [ShutdownReason](#) shutdownReason=[ShutdownReason.Ok](#), bool forceShutdownProcedure=false)
 - Initiates a [Simulation.Dispose](#).*
 - void [SinglePlayerContinue](#) ()
 - Continues a paused game in single player*
 - void [SinglePlayerPause](#) ()
 - Pauses the game in single player*
 - void [SinglePlayerPause](#) (bool paused)
 - Sets the paused state in a single player*
 - [NetworkObject](#) [Spawn](#) ([GameObject](#) prefab, [Vector3?](#) position=null, [Quaternion?](#) rotation=null, [PlayerRef?](#) inputAuthority=null, [OnBeforeSpawned](#) onBeforeSpawned=null, [NetworkSpawnFlags](#) flags=default)
 - Attempts to network instantiate a [NetworkObject](#) using a [GameObject](#). The supplied [GameObject](#) must have a [NetworkObject](#) component.*
 - [NetworkObject](#) [Spawn](#) ([NetworkObject](#) prefab, [Vector3?](#) position=null, [Quaternion?](#) rotation=null, [PlayerRef?](#) inputAuthority=null, [OnBeforeSpawned](#) onBeforeSpawned=null, [NetworkSpawnFlags](#) flags=default)
 - Attempts to network instantiate a [NetworkObject](#) using a [NetworkObject](#) prefab. Note: position and rotation values are only used locally for the instantiation of the object, and are not inherently networked. Use [NetworkTransform](#) (or any custom class derived from [NetworkTRSP](#)) to replicate the initial transform state.*

- [NetworkObject Spawn](#) ([NetworkObjectGuid](#) prefabGuid, [Vector3?](#) position=null, [Quaternion?](#) rotation=null, [PlayerRef?](#) inputAuthority=null, [OnBeforeSpawned](#) onBeforeSpawned=null, [NetworkSpawnFlags](#) flags=default)

Attempts to network instantiate a [NetworkObject](#) using a [NetworkObjectGuid](#) Note: position and rotation values are only used locally for the instantiation of the object, and are not inherently networked. Use [NetworkTransform](#) (or any custom class derived from [NetworkTRSP](#)) to replicate the initial transform state.
- [NetworkObject Spawn](#) ([NetworkPrefabId](#) typeId, [Vector3?](#) position=null, [Quaternion?](#) rotation=null, [PlayerRef?](#) inputAuthority=null, [OnBeforeSpawned](#) onBeforeSpawned=null, [NetworkSpawnFlags](#) flags=default)

Attempts to network instantiate a [NetworkObject](#) using a [NetworkPrefabId](#) Note: position and rotation values are only used locally for the instantiation of the object, and are not inherently networked. Use [NetworkTransform](#) (or any custom class derived from [NetworkTRSP](#)) to replicate the initial transform state.
- [NetworkObject Spawn](#) ([NetworkPrefabRef](#) prefabRef, [Vector3?](#) position=null, [Quaternion?](#) rotation=null, [PlayerRef?](#) inputAuthority=null, [OnBeforeSpawned](#) onBeforeSpawned=null, [NetworkSpawnFlags](#) flags=default)

Attempts to network instantiate a [NetworkObject](#) using a [NetworkPrefabRef](#). Note: position and rotation values are only used locally for the instantiation of the object, and are not inherently networked. Use [NetworkTransform](#) (or any custom class derived from [NetworkTRSP](#)) to replicate the initial transform state.
- [T Spawn< T >](#) ([T](#) prefab, [Vector3?](#) position=null, [Quaternion?](#) rotation=null, [PlayerRef?](#) inputAuthority=null, [OnBeforeSpawned](#) onBeforeSpawned=null, [NetworkSpawnFlags](#) flags=default)

Attempts to network instantiate a [NetworkObject](#) using a [Component](#) type that is part of a [NetworkObject](#)
- [NetworkSpawnOp SpawnAsync](#) ([GameObject](#) prefab, [Vector3?](#) position=null, [Quaternion?](#) rotation=null, [PlayerRef?](#) inputAuthority=null, [OnBeforeSpawned](#) onBeforeSpawned=null, [NetworkSpawnFlags](#) flags=default, [NetworkObjectSpawnDelegate](#) onCompleted=null)

Attempts to network instantiate a [NetworkObject](#) using a [GameObject](#). The supplied [GameObject](#) must have a [NetworkObject](#) component.
- [NetworkSpawnOp SpawnAsync](#) ([NetworkObject](#) prefab, [Vector3?](#) position=null, [Quaternion?](#) rotation=null, [PlayerRef?](#) inputAuthority=null, [OnBeforeSpawned](#) onBeforeSpawned=null, [NetworkSpawnFlags](#) flags=default, [NetworkObjectSpawnDelegate](#) onCompleted=null)

Attempts to network instantiate a [NetworkObject](#) using a [NetworkObject](#) prefab. Note: position and rotation values are only used locally for the instantiation of the object, and are not inherently networked. Use [NetworkTransform](#) (or any custom class derived from [NetworkTRSP](#)) to replicate the initial transform state.
- [NetworkSpawnOp SpawnAsync](#) ([NetworkObjectGuid](#) prefabGuid, [Vector3?](#) position=null, [Quaternion?](#) rotation=null, [PlayerRef?](#) inputAuthority=null, [OnBeforeSpawned](#) onBeforeSpawned=null, [NetworkSpawnFlags](#) flags=default, [NetworkObjectSpawnDelegate](#) onCompleted=null)

Attempts to network instantiate a [NetworkObject](#) using a [NetworkObjectGuid](#) Note: position and rotation values are only used locally for the instantiation of the object, and are not inherently networked. Use [NetworkTransform](#) (or any custom class derived from [NetworkTRSP](#)) to replicate the initial transform state.
- [NetworkSpawnOp SpawnAsync](#) ([NetworkPrefabId](#) typeId, [Vector3?](#) position=null, [Quaternion?](#) rotation=null, [PlayerRef?](#) inputAuthority=null, [OnBeforeSpawned](#) onBeforeSpawned=null, [NetworkSpawnFlags](#) flags=default, [NetworkObjectSpawnDelegate](#) onCompleted=null)

Attempts to network instantiate a [NetworkObject](#) using a [NetworkPrefabId](#) Note: position and rotation values are only used locally for the instantiation of the object, and are not inherently networked. Use [NetworkTransform](#) (or any custom class derived from [NetworkTRSP](#)) to replicate the initial transform state.
- [NetworkSpawnOp SpawnAsync](#) ([NetworkPrefabRef](#) prefabRef, [Vector3?](#) position=null, [Quaternion?](#) rotation=null, [PlayerRef?](#) inputAuthority=null, [OnBeforeSpawned](#) onBeforeSpawned=null, [NetworkSpawnFlags](#) flags=default, [NetworkObjectSpawnDelegate](#) onCompleted=null)

Attempts to network instantiate a [NetworkObject](#) using a [NetworkPrefabRef](#). Note: position and rotation values are only used locally for the instantiation of the object, and are not inherently networked. Use [NetworkTransform](#) (or any custom class derived from [NetworkTRSP](#)) to replicate the initial transform state.
- [NetworkSpawnOp SpawnAsync< T >](#) ([T](#) prefab, [Vector3?](#) position=null, [Quaternion?](#) rotation=null, [PlayerRef?](#) inputAuthority=null, [OnBeforeSpawned](#) onBeforeSpawned=null, [NetworkSpawnFlags](#) flags=default, [NetworkObjectSpawnDelegate](#) onCompleted=null)

Attempts to network instantiate a [NetworkObject](#) using a [Component](#) type that is part of a [NetworkObject](#)
- [Task< StartGameResult > StartGame](#) ([StartGameArgs](#) args)

Starts the local [Fusion](#) Runner and takes care of all major setup necessary

- `bool TryFindBehaviour (NetworkBehaviourId behaviourId, out NetworkBehaviour behaviour)`
Get the [NetworkBehaviour](#) instance for this [NetworkRunner](#) from a [NetworkBehaviourId](#).
- `bool TryFindBehaviour< T > (NetworkBehaviourId id, out T behaviour)`
Try to find a [NetworkBehaviour](#) with the provided [NetworkBehaviourId](#).
- `bool TryFindObject (NetworkId objectId, out NetworkObject networkObject)`
Get the [NetworkObject](#) instance for this [NetworkRunner](#) from a [NetworkId](#).
- `bool TryGetBehaviourStatistics (Type behaviourType, out BehaviourStatisticsSnapshot behaviourStatisticsSnapshot)`
Tries to get the statistics snapshot for a specified behaviour type.
- `bool TryGetFusionStatistics (out FusionStatisticsManager statisticsManager)`
Tries to get the [FusionStatisticsManager](#) from the simulation.
- `bool TryGetInputForPlayer< T > (PlayerRef player, out T input)`
Outputs the [NetworkInput](#) from player, translated to the indicated [INetworkInput](#).
- `T TryGetNetworkedBehaviourFromNetworkedObjectRef< T > (NetworkId networkId)`
Tries to return the first instance of T found on the root of a [NetworkObject](#).
- `NetworkBehaviourId TryGetNetworkedBehaviourId (NetworkBehaviour behaviour)`
Tries to return a [NetworkBehaviourId](#) for the [NetworkBehaviour](#) provided.
- `NetworkId TryGetObjectRefFromNetworkedBehaviour (NetworkBehaviour behaviour)`
Tries to return the behaviour [NetworkId](#).
- `bool TryGetPhysicsInfo (out NetworkPhysicsInfo info)`
Try to get the physics info.
- `bool TryGetPlayerObject (PlayerRef player, out NetworkObject networkObject)`
Try to gets the [NetworkObject](#) associated with a specific player
- `bool TryGetSceneInfo (out NetworkSceneInfo sceneInfo)`
Tries to get the [NetworkSceneInfo](#) of this [NetworkRunner](#).
- `bool TrySetPhysicsInfo (NetworkPhysicsInfo info)`
Try to set the physics info.
- `NetworkSpawnStatus TrySpawn (GameObject prefab, out NetworkObject obj, Vector3? position=null, Quaternion? rotation=null, PlayerRef? inputAuthority=null, OnBeforeSpawned onBeforeSpawned=null, NetworkSpawnFlags flags=default)`
Attempts to network instantiate a [NetworkObject](#) using a [GameObject](#). The supplied [GameObject](#) must have a [NetworkObject](#) component.
- `NetworkSpawnStatus TrySpawn (NetworkObject prefab, out NetworkObject obj, Vector3? position=null, Quaternion? rotation=null, PlayerRef? inputAuthority=null, OnBeforeSpawned onBeforeSpawned=null, NetworkSpawnFlags flags=default)`
Attempts to network instantiate a [NetworkObject](#) using a [NetworkObject](#) prefab. Note: position and rotation values are only used locally for the instantiation of the object, and are not inherently networked. Use [NetworkTransform](#) (or any custom class derived from [NetworkTRSP](#)) to replicate the initial transform state.
- `NetworkSpawnStatus TrySpawn (NetworkObjectGuid prefabGuid, out NetworkObject obj, Vector3? position=null, Quaternion? rotation=null, PlayerRef? inputAuthority=null, OnBeforeSpawned onBeforeSpawned=null, NetworkSpawnFlags flags=default)`
Attempts to network instantiate a [NetworkObject](#) using a [NetworkObjectGuid](#) Note: position and rotation values are only used locally for the instantiation of the object, and are not inherently networked. Use [NetworkTransform](#) (or any custom class derived from [NetworkTRSP](#)) to replicate the initial transform state.
- `NetworkSpawnStatus TrySpawn (NetworkPrefabId typeId, out NetworkObject obj, Vector3? position=null, Quaternion? rotation=null, PlayerRef? inputAuthority=null, OnBeforeSpawned onBeforeSpawned=null, NetworkSpawnFlags flags=default)`
Attempts to network instantiate a [NetworkObject](#) using a [NetworkPrefabId](#) Note: position and rotation values are only used locally for the instantiation of the object, and are not inherently networked. Use [NetworkTransform](#) (or any custom class derived from [NetworkTRSP](#)) to replicate the initial transform state.
- `NetworkSpawnStatus TrySpawn (NetworkPrefabRef prefabRef, out NetworkObject obj, Vector3? position=null, Quaternion? rotation=null, PlayerRef? inputAuthority=null, OnBeforeSpawned onBeforeSpawned=null, NetworkSpawnFlags flags=default)`

Attempts to network instantiate a [NetworkObject](#) using a [NetworkPrefabRef](#). Note: position and rotation values are only used locally for the instantiation of the object, and are not inherently networked. Use [NetworkTransform](#) (or any custom class derived from [NetworkTRSP](#)) to replicate the initial transform state.

- [NetworkSpawnStatus TrySpawn< T >](#) (T prefab, out T obj, Vector3? position=null, Quaternion? rotation=null, [PlayerRef?](#) inputAuthority=null, [OnBeforeSpawned](#) onBeforeSpawned=null, [NetworkSpawnFlags](#) flags=default)

Attempts to network instantiate a [NetworkObject](#) using a Component type that is part of a [NetworkObject](#)

- [NetworkSceneAsyncOp UnloadScene](#) ([SceneRef](#) sceneRef)
- [NetworkSceneAsyncOp UnloadScene](#) (string sceneName)

Unloads a scene

- void [UpdateInternal](#) (double dt)

This method is meant to be called by [INetworkRunnerUpdater](#).

Static Public Member Functions

- static [Task< List< RegionInfo > >](#) [GetAvailableRegions](#) (string appld=default, [CancellationToken](#) cancellationToken=default)

Starts an operation to retrieves the list of available regions.

- static [List< NetworkRunner >.Enumerator](#) [GetInstancesEnumerator](#) ()

Get enumerator for the collection of all [NetworkRunners](#). Allows to enumerate alloc-free.

- static [NetworkRunner](#) [GetRunnerForGameObject](#) ([GameObject](#) gameObject)

Get the [NetworkRunner](#) a [GameObject](#) instance belongs to.

- static [NetworkRunner](#) [GetRunnerForScene](#) ([Scene](#) scene)

Get the [NetworkRunner](#) from a specific [Scene](#)

Public Attributes

- [Func< string, ServerConnection, string >](#) [CloudAddressRewriter](#) = null

Static Public Attributes

- static [CloudConnectionLostHandler](#) [CloudConnectionLost](#)

Properties

- [IEnumerable< PlayerRef >](#) [ActivePlayers](#) [get]

Returns the collection of [PlayerRef](#) objects for this [NetworkRunner](#)'s [Fusion.Simulation](#).
- [AuthenticationValues](#) [AuthenticationValues](#) [get]

[AuthenticationValues](#) used by this [Runner](#) to Authenticate the local peer.
- static [BuildTypes](#) [BuildType](#) [get]

Get [Fusion.Runtime.dll](#) build type.
- bool [CanSpawn](#) [get]

Signal if the [Network Runner](#) can spawn a [NetworkObject](#)
- [NetworkProjectConfig](#) [Config](#) [get]

Returns the [NetworkProjectConfig](#) reference.
- [ConnectionType](#) [CurrentConnectionType](#) [get]

Check the current [Connection Type](#) with the Remote Server
- float [DeltaTime](#) [get]

Returns the fixed tick time interval. Derived from the [SimulationRuntimeConfig.TickRate](#).

- **GameMode** `GameMode` [get]
Current Game Mode active on the [Fusion Simulation](#)
- static `ReadOnlyList< NetworkRunner > Instances` [get]
A list of all [NetworkRunners](#).
- bool **IsClient** [get]
Returns if this [Fusion.Simulation](#) represents a Client connection.
- bool **IsCloudReady** [get]
Signal if the Local Peer is connected to Photon Cloud and is able to Create/Join Room but also receive Lobby Updates
- bool **IsConnectedToServer** [get]
Returns if this Client is currently connected to a Remote Server
- bool **IsFirstTick** [get]
If this is the first tick that executes this update or re-simulation
- bool **IsForward** [get]
If this is not a re-simulation but a new forward tick
- bool **IsInSession** [get]
- bool **IsLastTick** [get]
If this is the last tick that is being executed this update
- bool **IsPlayer** [get]
Returns true if this runner represents a Client or Host. Dedicated servers have no local player and will return false.
- bool **IsResimulation** [get]
If we are currently executing a client side prediction re-simulation.
- bool **IsResume** [get]
if this instance is a resume (host migration)
- bool **IsRunning** [get]
Returns if this [Fusion.Simulation](#) is valid and running.
- bool **IsSceneAuthority** [get]
Is this runner responsible for scene management.
- bool? **IsSceneManagerBusy** [get]
Signals if the [INetworkSceneManager](#) instance assigned to this [NetworkRunner](#) is busy with any scene loading operation.
- bool **IsServer** [get]
Returns if this [Fusion.Simulation](#) represents a Server connection.
- bool **IsSharedModeMasterClient** [get]
Signal if the Local Peer is in a Room and is the Room Master Client
- bool **IsShutdown** [get]
If the runner is shutdown
- bool **IsSimulationUpdating** [get]
Is the runner updating the simulation.
- bool **IsSinglePlayer** [get]
Returns true if this runner was started as single player (Started as [SimulationModes.Host](#) with [SimulationConfig.PlayerCount](#) = 1).
- bool **IsStarting** [get]
If the runner is pending to start
- **HitboxManager LagCompensation** [get]
Returns the global instance of a lag compensation buffer [Fusion.HitboxManager](#).
- **Tick LatestServerTick** [get]
Get the latest confirmed tick of the server we are aware of
- **LobbyInfo** `LobbyInfo = new LobbyInfo()` [get]
Signal if the local peer is already inside a Lobby
- float **LocalAlpha** [get]
Get the local time alpha value

- [PlayerRef LocalPlayer](#) [get]

Returns a [PlayerRef](#) for the local simulation. For a dedicated server [PlayerRef.IsRealPlayer](#) will equal false. [Player↔Refs](#) are assigned in order from 0 to [MaxPlayers-1](#) and are re-used as players join and leave. The only caveat is that the server player (if one exists), always gets the last index no matter how many clients are connected.
- float??? [LocalRenderTime](#) [get]

The current time (current [State.Time](#) + [Simulation.DeltaTime](#)) for predicted objects (objects in the local time frame). Use as an equivalent to Unity's [Time.time](#). Time is relative to [Tick 0](#) (which represents [Time 0f](#)).
- [SimulationModes Mode](#) [get]

Returns the [SimulationModes](#) flags for The type of network peer the associated [Fusion.Simulation](#) represents.
- [NATType NATType](#) [get]

Exposes the current NAT Type from the local Peer
- [INetworkObjectProvider ObjectProvider](#) [get]

Returns the [INetworkObjectProvider](#) instance.
- [NetworkPrefabTable Prefabs](#) [get]

Reference to the [NetworkPrefabTable](#).
- bool [ProvideInput](#) [get, set]

Indicates if this [NetworkRunner](#) is collecting [PlayerRef INetworkInput](#).
- int???? [ReliableDataSendRate](#) [get, set]
- float [RemoteRenderTime](#) [get]

The current time (current [State.Time](#) + [Simulation.DeltaTime](#)) for non-predicted objects (objects in a remote time frame). Use as an equivalent to Unity's [Time.time](#). Time is relative to [Tick 0](#) (which represents [Time 0f](#)).
- [INetworkSceneManager SceneManager](#) [get]

Returns the [INetworkSceneManager](#) instance.
- [SessionInfo SessionInfo = new SessionInfo\(\)](#) [get]

Stores information about the current running session
- float [SimulationTime](#) [get]

The time the current [State](#) represents (the most recent [FixedUpdateNetwork](#) simulation). Use as an equivalent to Unity's [Time.fixedTime](#). Time is relative to [Tick 0](#) (which represents [Time 0f](#)).
- Scene??? [SimulationUnityScene](#) [get]

The main scene of the [NetworkRunner](#) or default if not running.
- [SimulationStages Stage](#) [get]

Returns the current [SimulationStages](#) stage of this [Fusion.Simulation](#).
- [States State](#) [get]

The current state of the runner, if it's Starting, Running, Shutdown
- [Tick??? Tick](#) [get]

The tick associated with the current state of networked objects, or the current simulation tick being processed (when evaluated during [FixedUpdateNetwork](#)).
- int [TickRate](#) [get]
- int [TicksExecuted](#) [get]

Returns how many ticks we executed last update.
- [Topologies Topology](#) [get]

The current topology used
- string [UserId](#) [get]

Photon Client UserID

Events

- [ObjectDelegate ObjectAcquired](#)

Event for object acquired

6.250.1 Detailed Description

Host Migration related code in order to get a copy of the [Simulation](#) State

All Scene related API and fields

Represents a Server or Client [Simulation](#).

6.250.2 Member Enumeration Documentation

6.250.2.1 BuildTypes

```
enum BuildTypes [strong]
```

Enumeration of Fusion.Runtime.dll options.

Enumerator

Debug	Use the Debug version of the Fusion.Runtime.dll.
Release	Use the Debug version of the Fusion.Runtime.dll.

6.250.2.2 States

```
enum States [strong]
```

Initialization stages of [Fusion](#)

Enumerator

Starting	Runner is about to start
Running	Runner is running
Shutdown	Runner is shutdown

6.250.3 Member Function Documentation

6.250.3.1 AddCallbacks()

```
void AddCallbacks (
    params INetworkRunnerCallbacks[] callbacks )
```

Register an [INetworkRunnerCallbacks](#) instance for callbacks from this [NetworkRunner](#).

Parameters

<i>callbacks</i>	Callbacks to register
------------------	-----------------------

6.250.3.2 AddGlobal()

```
void AddGlobal (
    SimulationBehaviour instance )
```

Add and register a [SimulationBehaviour](#) to this [NetworkRunner](#). Note: It should NOT be a [NetworkBehaviour](#)

6.250.3.3 AddPlayerAreaOfInterest()

```
void AddPlayerAreaOfInterest (
    PlayerRef player,
    Vector3 center,
    float radius )
```

Call this every FixedUpdateNetwork to add an area of interest for a player. Should only be called from the Host/↔ Server in Server client mode. Should only be called for the local player in shared mode.

6.250.3.4 Attach() [1/2]

```
void Attach (
    NetworkObject networkObject,
    PlayerRef? inputAuthority = null,
    bool allocate = true,
    bool? masterClientObjectOverride = null )
```

Attaches a user-created network object to the network.

Parameters

<i>networkObject</i>	The network object to attach. Must not be null and must have a valid NetworkTypeId.
<i>inputAuthority</i>	Optional PlayerRef . If assigned, it will be the default input authority for this object.
<i>allocate</i>	Optional boolean. If true, the object will be allocated in memory and attached to the scene object. Default is true.
<i>masterClientObjectOverride</i>	Optional boolean. If provided, it will override the master client object setting. Default is null.

Exceptions

<i>ArgumentNullException</i>	Thrown when the provided network object is null.
<i>ArgumentException</i>	Thrown when the provided network object has an invalid NetworkTypeId.

6.250.3.5 Attach() [2/2]

```
void Attach (
    NetworkObject[] networkObjects,
    PlayerRef? inputAuthority = null,
    bool allocate = true,
    bool? masterClientObjectOverride = null )
```

Attach and assign to this [NetworkRunner](#) the [NetworkObject](#) provided. Used internally from the default implementation of [INetworkSceneManager](#) to register scene objects.

6.250.3.6 ClearPlayerAreaOfInterest()

```
void ClearPlayerAreaOfInterest (
    PlayerRef player )
```

Clears the area of interest for a player. This can only be called from the server/host

6.250.3.7 CloudConnectionLostHandler()

```
delegate void CloudConnectionLostHandler (
    NetworkRunner networkRunner,
    ShutdownReason shutdownReason,
    bool reconnecting )
```

If this callback is implemented, the default behavior of disconnecting when the cloud connection is lost in server/client or host/client mode is disabled. This callback will be invoked instead, leaving the decision up to the user. It is called both on the server/host and on the client.

Parameters

<i>networkRunner</i>	NetworkRunner instance
<i>shutdownReason</i>	Shutdown reason for the connection loss
<i>reconnecting</i>	Flag to signal if the peer is attempting to reconnect to the Photon Cloud

6.250.3.8 Despawn()

```
void Despawn (
    NetworkObject networkObject )
```

Destroys a [NetworkObject](#).

Parameters

<i>networkObject</i>	The NetworkObject to be destroyed.
----------------------	--

This method checks if the local simulation has state authority over the [NetworkObject](#). If it does, it checks if the [NetworkObject](#) exists and has state authority. If these conditions are met, it destroys the [NetworkObject](#).

Exceptions

<i>InvalidOperationException</i>	Thrown when the NetworkObject does not belong to this runner.
----------------------------------	---

6.250.3.9 DestroySingleton< T >()

```
void DestroySingleton< T > ( )
```

Removes a specific [SimulationBehaviour](#) from this [NetworkRunner](#) gameobject, if it exists.

Type Constraints

T: [SimulationBehaviour](#)

6.250.3.10 Disconnect()

```
void Disconnect (
    PlayerRef player,
    byte[] token = null )
```

Disconnects a player from the server.

Parameters

<i>player</i>	The player to disconnect. Must be a valid PlayerRef .
<i>token</i>	Optional byte array. If provided, it will be used as the disconnection token.

This method can only be called from the server. If called from a client, an error message will be logged.

6.250.3.11 EnsureRunnerSceneIsActive()

```
bool EnsureRunnerSceneIsActive (
    out Scene previousActiveScene )
```

Ensures the scene of this runner is active and returns the previous active scene

Parameters

<i>previousActiveScene</i>	Previous active scene
----------------------------	-----------------------

Returns

True if the scene was changed, false otherwise

6.250.3.12 Exists() [1/2]

```
bool Exists (
    NetworkId id )
```

Returns if the [Fusion.Simulation](#) contains a [NetworkObject](#) with given *id* in the current State.

6.250.3.13 Exists() [2/2]

```
bool Exists (
    NetworkObject obj )
```

Returns if the [Fusion.Simulation](#) contains a reference to a [NetworkObject](#) in the current State.

6.250.3.14 FindObject()

```
NetworkObject FindObject (
    NetworkId networkId )
```

Get the [NetworkObject](#) instance for this [NetworkRunner](#) from a [NetworkId](#).

Parameters

<i>networkId</i>	NetworkID to look forward
------------------	---------------------------

Returns

null if object cannot be found.

6.250.3.15 GetAllBehaviours()

```
SimulationBehaviour [] GetAllBehaviours (
    Type type )
```

Returns an array of all [SimulationBehaviour](#) instances registered with this [NetworkRunner](#).

Parameters

<i>type</i>	The type of the behaviours to be returned.
-------------	--

Returns

An array of [SimulationBehaviour](#) instances of the specified type.

6.250.3.16 GetAllBehaviours< T >() [1/2]

```
List<T> GetAllBehaviours< T > ( )
```

Get a list with all behaviours of the desired type that are registered on the [NetworkRunner](#).

Template Parameters

<i>T</i>	SimulationBehaviour type
----------	--

Returns

The result list

Type Constraints

T : [SimulationBehaviour](#)

6.250.3.17 GetAllBehaviours< T >() [2/2]

```
void GetAllBehaviours< T > (
    List< T > result )
```

Add on the list all behaviours of the desired type that are registered on the [NetworkRunner](#). Note: The list will not be cleared before adding the results.

Parameters

<i>result</i>	The list to add the behaviours
---------------	--------------------------------

Template Parameters

<i>T</i>	SimulationBehaviour type
----------	--

Type Constraints

***T* : [SimulationBehaviour](#)**

6.250.3.18 GetAllNetworkObjects() [1/2]

```
List<NetworkObject> GetAllNetworkObjects ( )
```

Retrieves a list of all network objects in the simulation.

Returns

A list of [NetworkObject](#) instances.

6.250.3.19 GetAllNetworkObjects() [2/2]

```
void GetAllNetworkObjects (
    List< NetworkObject > result )
```

Populate a list with all network objects in the simulation.

Parameters

<i>result</i>	The list to which the network objects will be added.
---------------	--

6.250.3.20 GetAreaOfInterestGizmoData()

```
void GetAreaOfInterestGizmoData (
    List<(Vector3 center, Vector3 size, int playerCount, int objectCount)> result )
```

Populates the provided list with data about the current Area of Interest (AOI) cells. Each element in the list represents one AOI cell.

Parameters

<i>result</i>	The list to be populated with AOI cell data. Each tuple in the list contains the center of the AOI cell, its size, the count of players in the cell, and the count of objects in the cell.
---------------	--

6.250.3.21 GetAvailableRegions()

```
static Task<List<RegionInfo> > GetAvailableRegions (
    string appId = default,
    CancellationToken cancellationToken = default ) [static]
```

Starts an operation to retrieves the list of available regions.

Parameters

<i>appId</i>	Optional App ID. If not provided, the method will use the global App ID from the PhotonAppSettings.
<i>cancellationToken</i>	Optional CancellationToken parameter that can be used to cancel the operation.

Returns

Returns a list with information about all the available regions.

6.250.3.22 GetInputForPlayer< T >()

```
T? GetInputForPlayer< T > (
    PlayerRef player )
```

Returns the [NetworkInput](#) data from player, converted to the indicated [INetworkInput](#).

Type Constraints

T* : *unmanaged

T* : *INetworkInput

6.250.3.23 GetInstancesEnumerator()

```
static List<NetworkRunner>.Enumerator GetInstancesEnumerator ( ) [static]
```

Get enumerator for the collection of all [NetworkRunners](#). Allows to enumerate alloc-free.

6.250.3.24 GetInterfaceListHead()

```
SimulationBehaviourListScope GetInterfaceListHead (  
    Type type,  
    int index,  
    out SimulationBehaviour head )
```

Get the interface list head.

Parameters

<i>type</i>	The interface type
<i>index</i>	The desired index on the list of behaviourList
<i>head</i>	The head reference

Returns

A disposable [SimulationBehaviourListScope](#) to be used on an `using` scope

6.250.3.25 GetInterfaceListNext()

```
SimulationBehaviour GetInterfaceListNext (
    SimulationBehaviour behaviour )
```

Get the next behaviour

Parameters

<i>behaviour</i>	The reference behaviour to get the next one
------------------	---

Returns

Gives the next behaviour

6.250.3.26 GetInterfaceListPrev()

```
SimulationBehaviour GetInterfaceListPrev (
    SimulationBehaviour behaviour )
```

Get the previous behaviour

Parameters

<i>behaviour</i>	The reference behaviour to get the previous one
------------------	---

Returns

Gives the previous behaviour

6.250.3.27 GetInterfaceListsCount()

```
int GetInterfaceListsCount (
    Type type )
```

Get the number of interfaces of the desired type that are registered on the behaviour updater.

Parameters

<i>type</i>	The interface type
-------------	--------------------

Returns

The number of interfaces

6.250.3.28 GetMemorySnapshot()

```
void GetMemorySnapshot (
    MemoryStatisticsSnapshot.TargetAllocator targetAllocator,
    ref MemoryStatisticsSnapshot snapshot )
```

Gets a memory snapshot of the simulation.

Parameters

<i>targetAllocator</i>	The target allocator for the memory statistics. Must be a valid value from the MemoryStatisticsSnapshot.TargetAllocator enum.
<i>snapshot</i>	A reference to the MemoryStatisticsSnapshot struct that will store the memory snapshot.

6.250.3.29 GetObjectsInAreaOfInterestForPlayer()

```
List<NetworkId> GetObjectsInAreaOfInterestForPlayer (
    PlayerRef player )
```

Retrieves a list of network object IDs that are in the area of interest for the specified player. Server only.

Parameters

<i>player</i>	The player for whom the area of interest is being queried.
---------------	--

Returns

A list of network object IDs in the area of interest for the player.

6.250.3.30 GetPhysicsScene()

```
PhysicsScene GetPhysicsScene ( )
```

Get the 3D Physics scene being used by this Runner.

6.250.3.31 GetPhysicsScene2D()

```
PhysicsScene2D GetPhysicsScene2D ( )
```

Get the 2D Physics scene being used by this Runner.

6.250.3.32 GetPlayerActorId()

```
int? GetPlayerActorId (
    PlayerRef player )
```

Gets Player's Actor Number (ID).

If used in Shared Mode, every client can get this information. If used in Client Server Mode, only the Server is able to get this information.

Parameters

<i>player</i>	PlayerRef to get the Actor Number (ID)
---------------	--

Returns

Actor Number associated with the [PlayerRef](#), otherwise null.

6.250.3.33 GetPlayerConnectionToken()

```
byte [] GetPlayerConnectionToken (
    PlayerRef player = default )
```

Returns a copy of the Connection Token used by a Player when connecting to this Server. Only available on Server. It will return null if running on a Client or the Connection token is missing

Parameters

<i>player</i>	PlayerRef to check for a Connection Token
---------------	---

Returns

Copy of the Connection Token

6.250.3.34 GetPlayerConnectionType()

```
ConnectionType GetPlayerConnectionType (  
    PlayerRef player )
```

Return the [ConnectionType](#) with a Remote [PlayerRef](#). Valid only when invoked from a Server ([NetworkRunner.IsServer](#))

Parameters

<i>player</i>	Remote Player to check the ConnectionType
---------------	---

Returns

[ConnectionType](#) with a [PlayerRef](#)

6.250.3.35 GetPlayerObject()

```
NetworkObject GetPlayerObject (  
    PlayerRef player )
```

Gets the network object associated with a specific player

Parameters

<i>player</i>	PlayerRef to get the network object
---------------	---

Returns

Network object if one is associated with the player

6.250.3.36 GetPlayerRtt()

```
double GetPlayerRtt (  
    PlayerRef playerRef )
```

Returns the player round trip time (ping) in seconds

Parameters

<i>playerRef</i>	The player you want the round trip time for
------------------	---

6.250.3.37 GetPlayerUserId()

```
string GetPlayerUserId (
    PlayerRef player = default )
```

Gets Player's UserID.

If used in Shared Mode, every client can get this information. If used in Client Server Mode, only the Server is able to get this information.

Parameters

<i>player</i>	PlayerRef to get the UserID. If no PlayerRef is passed, the UserID of the local client is returned instead.
---------------	---

Returns

UserID if valid player found, otherwise null.

6.250.3.38 GetRawInputForPlayer()

```
NetworkInput? GetRawInputForPlayer (
    PlayerRef player )
```

Returns the unconverted unsafe [NetworkInput](#) for the indicated player.

6.250.3.39 GetResumeSnapshotNetworkObjects()

```
IEnumerable<NetworkObject> GetResumeSnapshotNetworkObjects ( )
```

Iterate over the old NetworkObjects from the Resume Snapshot

Returns

Iterable list of [NetworkObject](#)

6.250.3.40 GetResumeSnapshotNetworkSceneObjects()

```
IEnumerable<NetworkObject, NetworkObjectHeaderPtr> GetResumeSnapshotNetworkSceneObjects ( )
```

Iterate over the Scene NetworkObjects from the Resume Snapshot while giving the reference of the old Snapshot data associated with that particular Scene Object

Returns

Iterable list of Scene [NetworkObject](#) and Scene Object Header

6.250.3.41 GetRpcTargetStatus()

```
RpcTargetStatus GetRpcTargetStatus (
    PlayerRef target )
```

Return the [RpcTargetStatus](#) for a specific player.

6.250.3.42 GetRunnerForGameObject()

```
static NetworkRunner GetRunnerForGameObject (
    GameObject gameObject ) [static]
```

Get the [NetworkRunner](#) a GameObject instance belongs to.

Parameters

<i>gameObject</i>	GameObject to check for a NetworkRunner
-------------------	---

Returns

[NetworkRunner](#) reference, or null if not found

6.250.3.43 GetRunnerForScene()

```
static NetworkRunner GetRunnerForScene (
    Scene scene ) [static]
```

Get the [NetworkRunner](#) from a specific Scene

Parameters

<i>scene</i>	Scene to check for a NetworkRunner
--------------	--

Returns

[NetworkRunner](#) reference, or null if not found

6.250.3.44 GetSingleton< T >()

```
T GetSingleton< T > ( )
```

Ensures that a specific [SimulationBehaviour](#) component exists on this [NetworkRunner](#) gameobject.

Type Constraints

T : SimulationBehaviour

6.250.3.45 HasSingleton< T >()

```
bool HasSingleton< T > ( )
```

Returns if a given [SimulationBehaviour](#) is present in this [NetworkRunner](#) gameobject.

Returns

Returns true if the [SimulationBehaviour](#) was found

Type Constraints

T : SimulationBehaviour

6.250.3.46 InstantiateInRunnerScene() [1/2]

```
GameObject InstantiateInRunnerScene (
    GameObject original )
```

Instantiates an object in the scene of this runner

6.250.3.47 InstantiateInRunnerScene() [2/2]

```
GameObject InstantiateInRunnerScene (
    GameObject original,
    Vector3 position,
    Quaternion rotation )
```

Instantiates an object in the scene of this runner

6.250.3.48 InstantiateInRunnerScene< T >() [1/2]

```
T InstantiateInRunnerScene< T > (
    T original )
```

Instantiates an object in the scene of this runner

Type Constraints

***T* : Component**

6.250.3.49 InstantiateInRunnerScene< T >() [2/2]

```
T InstantiateInRunnerScene< T > (
    T original,
    Vector3 position,
    Quaternion rotation )
```

Instantiates an object in the scene of this runner

Type Constraints

***T* : Component**

6.250.3.50 InvokeSceneLoadDone()

```
void InvokeSceneLoadDone (
    in SceneLoadDoneArgs info )
```

Invoke [INetworkRunnerCallbacks.OnSceneLoadDone\(NetworkRunner\)](#) on all implementations

6.250.3.51 InvokeSceneLoadStart()

```
void InvokeSceneLoadStart (
    SceneRef sceneRef )
```

Invoke [INetworkRunnerCallbacks.OnSceneLoadStart\(NetworkRunner\)](#) on all implementations

6.250.3.52 IsInterestedIn()

```
bool? IsInterestedIn (
    NetworkObject obj,
    PlayerRef player )
```

Test if a player has Interest in a [NetworkObject](#).

Returns

Returns null if interest cannot be determined (clients without State Authority are not aware of other client's Object Interest)

6.250.3.53 IsPlayerValid()

```
bool IsPlayerValid (
    PlayerRef player )
```

Checks if the provided player is valid in the current simulation.

Parameters

<i>player</i>	The player reference to be validated.
---------------	---------------------------------------

Returns

Returns true if the player is valid, false otherwise.

6.250.3.54 JoinSessionLobby()

```
async Task<StartGameResult> JoinSessionLobby (
    SessionLobby sessionLobby,
    string lobbyID = null,
    AuthenticationValues authentication = null,
    FusionAppSettings customAppSettings = null,
    bool? useDefaultCloudPorts = false,
    CancellationToken cancellationToken = default,
    bool useCachedRegions = true )
```

Join the Peer to a specific Lobby, either a prebuild or a custom one.

More about matchmaking: <https://doc.photonengine.com/en-us/fusion/current/manual/matchmaking>

Parameters

<i>sessionLobby</i>	Lobby Type to Join
<i>lobbyID</i>	Lobby ID

Parameters

<i>authentication</i>	Authentication Values used to authenticate this peer
<i>customAppSettings</i>	Custom Photon Application Settings
<i>useDefaultCloudPorts</i>	Signal if the LoadBalancingClient should use the Default or Alternative Ports
<i>cancellationToken</i>	Optional Cancellation Token
<i>useCachedRegions</i>	Signal if the cached regions ping should be used to speed up connection

Returns

Async Task to Join a Session Lobby. Can be used to wait for the process to be finished.

6.250.3.55 LoadScene() [1/3]

```
NetworkSceneAsyncOp LoadScene (
    SceneRef sceneRef,
    LoadSceneMode loadSceneMode = LoadSceneMode.Single,
    LocalPhysicsMode localPhysicsMode = LocalPhysicsMode.None,
    bool setActiveOnLoad = DefaultSetActiveOnLoad )
```

Loads a scene

Parameters

<i>sceneRef</i>	Reference to the scene to load
<i>loadSceneMode</i>	Scene load mode
<i>localPhysicsMode</i>	Scene physics mode
<i>setActiveOnLoad</i>	Should the scene be set as active when loaded

Returns

Scene Load operation

6.250.3.56 LoadScene() [2/3]

```
NetworkSceneAsyncOp LoadScene (
    string sceneName,
    LoadSceneMode loadSceneMode = LoadSceneMode.Single,
    LocalPhysicsMode localPhysicsMode = LocalPhysicsMode.None,
    bool setActiveOnLoad = DefaultSetActiveOnLoad )
```

Loads a scene

Parameters

<i>sceneName</i>	Name of the scene to load
<i>loadSceneMode</i>	Scene load mode
<i>localPhysicsMode</i>	Scene physics mode
<i>setActiveOnLoad</i>	Should the scene be set as active when loaded

Returns

Scene Load operation

6.250.3.57 LoadScene() [3/3]

```
NetworkSceneAsyncOp LoadScene (
    string sceneName,
    LoadSceneParameters parameters,
    bool setActiveOnLoad = DefaultSetActiveOnLoad )
```

Loads a scene

Parameters

<i>sceneName</i>	Name of the scene to load
<i>parameters</i>	Parameters to use when loading the scene
<i>setActiveOnLoad</i>	Should the scene be set as active when loaded

Returns

Scene Load operation

6.250.3.58 MakeDontDestroyOnLoad()

```
void MakeDontDestroyOnLoad (
    GameObject obj )
```

Mark an object as DontDestroyOnLoad.

Parameters

<i>obj</i>	Object to mark
------------	----------------

6.250.3.59 MoveGameObjectToSameScene()

```
bool MoveGameObjectToSameScene (
    GameObject gameObj,
    GameObject other )
```

Moves a GameObject to the same scene as another GameObject

Parameters

<i>gameObj</i>	Game Object to move
<i>other</i>	Game Object to move to the same scene as

Returns

True if the object was moved, false otherwise

6.250.3.60 MoveGameObjectToScene()

```
bool MoveGameObjectToScene (
    GameObject gameObj,
    SceneRef sceneRef )
```

Moves a GameObject to a specific scene

Parameters

<i>gameObj</i>	Game Object to move
<i>sceneRef</i>	Scene to move the object to

Returns

True if the object was moved, false otherwise

6.250.3.61 MoveToRunnerScene()

```
void MoveToRunnerScene (
    GameObject instance,
    SceneRef? targetSceneRef = null )
```

Moves an object to the scene of this runner

Parameters

<i>instance</i>	Object to move
<i>targetSceneRef</i>	Target scene to move the object to

6.250.3.62 MoveToRunnerScene< T >()

```
void MoveToRunnerScene< T > (
    T component )
```

Moves an object to the scene of this runner

Template Parameters

<i>T</i>	
----------	--

Parameters

<i>component</i>	Component of object to move
------------------	-----------------------------

Type Constraints

T* : *Component

6.250.3.63 ObjectDelegate()

```
delegate void ObjectDelegate (
    NetworkRunner runner,
    NetworkObject obj )
```

Delegate type for object callback

6.250.3.64 OnBeforeSpawned()

```
delegate void OnBeforeSpawned (
    NetworkRunner runner,
    NetworkObject obj )
```

Delegate type for on before spawned callback

6.250.3.65 PushHostMigrationSnapshot()

```
async Task<bool> PushHostMigrationSnapshot ( )
```

Compute and send a Host Migration Snapshot to the Photon Cloud

Returns

Task with the result of the operation. True if it was successful, false otherwise.

6.250.3.66 RegisterSceneObjects()

```
int RegisterSceneObjects (
    SceneRef scene,
    NetworkObject[] objects,
    NetworkSceneLoadId loadId = default )
```

Registers scene objects to the network.

Parameters

<i>scene</i>	The scene reference. Must be valid.
<i>objects</i>	Array of NetworkObject instances to be registered. Must not be null.
<i>loadId</i>	Optional NetworkSceneLoadId . Default value is used if not provided.

Returns

The number of objects registered.

Exceptions

<i>ArgumentException</i>	Thrown when the provided scene is not valid.
<i>ArgumentNullException</i>	Thrown when the provided objects array is null.

6.250.3.67 ReleaseStateAuthority()

```
void ReleaseStateAuthority (
    NetworkId id )
```

Releases state authority for a given [NetworkId](#) on shared mode.

Parameters

<i>id</i>	The NetworkId for which state authority needs to be released.
-----------	---

6.250.3.68 RemoveCallbacks()

```
void RemoveCallbacks (
    params INetworkRunnerCallbacks[] callbacks )
```

Unregister an [INetworkRunnerCallbacks](#) instance for callbacks from this [NetworkRunner](#).

Parameters

<i>callbacks</i>	Callbacks to unregister
------------------	-------------------------

6.250.3.69 RemoveGlobal()

```
void RemoveGlobal (
    SimulationBehaviour instance )
```

Removes a specific [SimulationBehaviour](#) from this [NetworkObject](#) gameobject, if it exists.

6.250.3.70 RenderInternal()

```
void RenderInternal ( )
```

This method is meant to be called by [INetworkRunnerUpdater](#).

6.250.3.71 RequestStateAuthority()

```
void RequestStateAuthority (
    NetworkId id )
```

Requests state authority for a specified [NetworkId](#) on shared mode.

Parameters

<i>id</i>	The NetworkId for which state authority is being requested.
-----------	---

6.250.3.72 SendReliableDataToPlayer()

```
void SendReliableDataToPlayer (
    PlayerRef player,
    ReliableKey key,
    byte[] data )
```

Sends a reliable data buffer to a target player.

Parameters

<i>player</i>	The player who should receive the buffer.
<i>key</i>	The key associated with the reliable data.
<i>data</i>	The data buffer to be sent.

6.250.3.73 SendReliableDataToServer()

```
void SendReliableDataToServer (
    ReliableKey key,
    byte[] data )
```

Sends a reliable data buffer to the server.

Parameters

<i>key</i>	The key associated with the reliable data.
<i>data</i>	The data buffer to be sent.

If the runner is a client, the data is sent to the server (connection index 0) with the player's index. If the runner is a server, the data is sent via the simulation callbacks.

6.250.3.74 SendRpc() [1/2]

```
void SendRpc (
    SimulationMessage * message )
```

Sends RPC message. Not meant to be used directly, ILWeaver calls this.

Parameters

<i>message</i>	SimulationMessage to send
----------------	---

6.250.3.75 SendRpc() [2/2]

```
void SendRpc (
    SimulationMessage * message,
    out RpcSendResult info )
```

Sends RPC message. Not meant to be used directly, ILWeaver calls this.

Parameters

<i>message</i>	SimulationMessage to send
<i>info</i>	RpcSendResult

6.250.3.76 SetAreaOfInterestCellSize()

```
void SetAreaOfInterestCellSize (
    int size )
```

Set the area of interest cell size

Parameters

<i>size</i>	Size of the cell
-------------	------------------

Exceptions

<i>Exception</i>	Can't change cell size in shared mode
------------------	---------------------------------------

6.250.3.77 SetAreaOfInterestGrid()

```
void SetAreaOfInterestGrid (
    int x,
    int y,
    int z )
```

Set the area of interest grid dimensions

Parameters

<i>x</i>	X dimension
<i>y</i>	Y dimension
<i>z</i>	Z dimension

Exceptions

<i>Exception</i>	Can't change grid size in shared mode
------------------	---------------------------------------

6.250.3.78 SetBehaviourReplicateTo()

```
void SetBehaviourReplicateTo (
    NetworkBehaviour behaviour,
    PlayerRef player,
    bool replicate )
```

Controls if a specific network behaviours state is replicated to a player or not

Parameters

<i>behaviour</i>	The behaviour to change replication status for
<i>player</i>	The player to change replication status for
<i>replicate</i>	true = replicate, false = don't replicate

6.250.3.79 SetBehaviourReplicateToAll()

```
void SetBehaviourReplicateToAll (
    NetworkBehaviour behaviour,
    bool replicate )
```

Controls if a specific network behaviours state is replicated to all players or not

Parameters

<i>behaviour</i>	The behaviour to change replication status for
<i>replicate</i>	true = replicate, false = don't replicate

6.250.3.80 SetIsSimulated()

```
bool SetIsSimulated (
    NetworkObject obj,
    bool simulate )
```

Sets the simulation state for this object, if it takes part in the NetworkFixedUpdate, etc. In shared mode a client cannot change the simulation state of [NetworkObjects](#) it does not have state authority over.

Parameters

<i>obj</i>	the object to change state for
<i>simulate</i>	true if it should be simulated, false if otherwise

Returns

true if the state of the object changed, false otherwise

6.250.3.81 SetMasterClient()

```
void SetMasterClient (
    PlayerRef player )
```

Promote a player to be the new master client. Only the master client is able to call this method

Parameters

<i>player</i>	The player to be promoted to master client
---------------	--

6.250.3.82 SetPlayerAlwaysInterested()

```
void SetPlayerAlwaysInterested (
    PlayerRef player,
    NetworkObject networkObject,
    bool alwaysInterested )
```

Flags this player as always interested in this object. Means it does not have to be in a players area of interest to be replicated. Only the [NetworkObject](#) State Authority can set interest.

Parameters

<i>player</i>	The player
<i>networkObject</i>	The object
<i>alwaysInterested</i>	If he's always interested, or not.

6.250.3.83 SetPlayerObject()

```
void SetPlayerObject (
    PlayerRef player,
    NetworkObject networkObject )
```

Sets the network object associated with this player

Parameters

<i>player</i>	PlayerRef to set the network object
<i>networkObject</i>	Network object to associate with the player

6.250.3.84 SetSimulateMultiPeerPhysics()

```
void SetSimulateMultiPeerPhysics (
    bool value )
```

Set the value for simulating physics scenes when using multi-peer. Restores the default physics simulation settings if false, overrides the default if true. (2D: [SimulationMode2D.Script](#) | 3D: [AutoSimulation](#) = false)

Parameters

<i>value</i>	The value to set. True to simulate physics scenes, false otherwise.
--------------	---

6.250.3.85 Shutdown()

```
Task Shutdown (
    bool destroyGameObject = true,
    ShutdownReason shutdownReason = ShutdownReason.Ok,
    bool forceShutdownProcedure = false )
```

Initiates a Simulation.Dispose.

6.250.3.86 SinglePlayerContinue()

```
void SinglePlayerContinue ( )
```

Continues a paused game in single player

6.250.3.87 SinglePlayerPause() [1/2]

```
void SinglePlayerPause ( )
```

Pauses the game in single player

6.250.3.88 SinglePlayerPause() [2/2]

```
void SinglePlayerPause (
    bool paused )
```

Sets the paused state in a single player

6.250.3.89 Spawn() [1/5]

```
NetworkObject Spawn (
    GameObject prefab,
    Vector3? position = null,
    Quaternion? rotation = null,
    PlayerRef? inputAuthority = null,
    OnBeforeSpawned onBeforeSpawned = null,
    NetworkSpawnFlags flags = default )
```

Attempts to network instantiate a [NetworkObject](#) using a [GameObject](#). The supplied [GameObject](#) must have a [NetworkObject](#) component.

Parameters

<i>prefab</i>	A GameObject with a NetworkObject
<i>position</i>	Spawn Position
<i>rotation</i>	Spawn Rotation
<i>inputAuthority</i>	Player Input Authority
<i>onBeforeSpawned</i>	OnBeforeSpawned reference
<i>flags</i>	Spawn flags

Returns

[NetworkObject](#) reference, or null if it was not able to spawn the object

6.250.3.90 Spawn() [2/5]

```
NetworkObject Spawn (
    NetworkObject prefab,
    Vector3? position = null,
    Quaternion? rotation = null,
    PlayerRef? inputAuthority = null,
    OnBeforeSpawned onBeforeSpawned = null,
    NetworkSpawnFlags flags = default )
```

Attempts to network instantiate a [NetworkObject](#) using a [NetworkObject](#) prefab. Note: position and rotation values are only used locally for the instantiation of the object, and are not inherently networked. Use [NetworkTransform](#) (or any custom class derived from [NetworkTRSP](#)) to replicate the initial transform state.

Parameters

<i>prefab</i>	Prefab used to spawn the NetworkObject
<i>position</i>	Spawn Position
<i>rotation</i>	Spawn Rotation
<i>inputAuthority</i>	Player Input Authority
<i>onBeforeSpawned</i>	OnBeforeSpawned reference
<i>flags</i>	Spawn flags

Returns

[NetworkObject](#) reference, or null if it was not able to spawn the object

6.250.3.91 Spawn() [3/5]

```
NetworkObject Spawn (
    NetworkObjectGuid prefabGuid,
```



```

Vector3? position = null,
Quaternion? rotation = null,
PlayerRef? inputAuthority = null,
OnBeforeSpawned onBeforeSpawned = null,
NetworkSpawnFlags flags = default )

```

Attempts to network instantiate a [NetworkObject](#) using a [NetworkObjectGuid](#) Note: position and rotation values are only used locally for the instantiation of the object, and are not inherently networked. Use [NetworkTransform](#) (or any custom class derived from [NetworkTRSP](#)) to replicate the initial transform state.

Parameters

<i>prefabGuid</i>	Object Guid used to spawn the NetworkObject
<i>position</i>	Spawn Position
<i>rotation</i>	Spawn Rotation
<i>inputAuthority</i>	Player Input Authority
<i>onBeforeSpawned</i>	OnBeforeSpawned reference
<i>flags</i>	Spawn flags

Returns

[NetworkObject](#) reference, or null if it was not able to spawn the object

6.250.3.92 Spawn() [4/5]

```

NetworkObject Spawn (
    NetworkPrefabId typeId,
    Vector3? position = null,
    Quaternion? rotation = null,
    PlayerRef? inputAuthority = null,
    OnBeforeSpawned onBeforeSpawned = null,
    NetworkSpawnFlags flags = default )

```

Attempts to network instantiate a [NetworkObject](#) using a [NetworkPrefabId](#) Note: position and rotation values are only used locally for the instantiation of the object, and are not inherently networked. Use [NetworkTransform](#) (or any custom class derived from [NetworkTRSP](#)) to replicate the initial transform state.

Parameters

<i>typeId</i>	Prefab ID used to spawn the NetworkObject
<i>position</i>	Spawn Position
<i>rotation</i>	Spawn Rotation
<i>inputAuthority</i>	Player Input Authority
<i>onBeforeSpawned</i>	OnBeforeSpawned reference
<i>flags</i>	Spawn flags

Returns

[NetworkObject](#) reference, or null if it was not able to spawn the object

6.250.3.93 Spawn() [5/5]

```

NetworkObject Spawn (
    NetworkPrefabRef prefabRef,
    Vector3? position = null,
    Quaternion? rotation = null,
    PlayerRef? inputAuthority = null,
    OnBeforeSpawned onBeforeSpawned = null,
    NetworkSpawnFlags flags = default )

```

Attempts to network instantiate a [NetworkObject](#) using a [NetworkPrefabRef](#). Note: position and rotation values are only used locally for the instantiation of the object, and are not inherently networked. Use [NetworkTransform](#) (or any custom class derived from [NetworkTRSP](#)) to replicate the initial transform state.

Parameters

<i>prefabRef</i>	Prefab Ref used to spawn the NetworkObject
<i>position</i>	Spawn Position
<i>rotation</i>	Spawn Rotation
<i>inputAuthority</i>	Player Input Authority
<i>onBeforeSpawned</i>	OnBeforeSpawned reference
<i>flags</i>	Spawn flags

Returns

[NetworkObject](#) reference, or null if it was not able to spawn the object

6.250.3.94 Spawn< T >()

```

T Spawn< T > (
    T prefab,
    Vector3? position = null,
    Quaternion? rotation = null,
    PlayerRef? inputAuthority = null,
    OnBeforeSpawned onBeforeSpawned = null,
    NetworkSpawnFlags flags = default )

```

Attempts to network instantiate a [NetworkObject](#) using a Component type that is part of a [NetworkObject](#)

Template Parameters

<i>T</i>	Must be a Type derived from SimulationBehaviour
----------	---

Parameters

<i>prefab</i>	SimulationBehaviour used to spawn the NetworkObject
---------------	---

Returns

T reference, or null if it was not able to spawn the object

Parameters

<i>position</i>	Spawn Position
<i>rotation</i>	Spawn Rotation
<i>inputAuthority</i>	Player Input Authority
<i>onBeforeSpawned</i>	OnBeforeSpawned reference
<i>flags</i>	Spawn flags

Type Constraints

T : *SimulationBehaviour*

6.250.3.95 SpawnAsync() [1/5]

```
NetworkSpawnOp SpawnAsync (
    GameObject prefab,
    Vector3? position = null,
    Quaternion? rotation = null,
    PlayerRef? inputAuthority = null,
    OnBeforeSpawned onBeforeSpawned = null,
    NetworkSpawnFlags flags = default,
    NetworkObjectSpawnDelegate onCompleted = null )
```

Attempts to network instantiate a [NetworkObject](#) using a `GameObject`. The supplied `GameObject` must have a [NetworkObject](#) component.

Parameters

<i>prefab</i>	A <code>GameObject</code> with a NetworkObject
<i>position</i>	Spawn Position
<i>rotation</i>	Spawn Rotation
<i>inputAuthority</i>	Player Input Authority
<i>onBeforeSpawned</i>	OnBeforeSpawned reference
<i>flags</i>	Spawn flags

,

Parameters

<i>onCompleted</i>	A callback to fire once the spawn is done.
--------------------	--

6.250.3.96 SpawnAsync() [2/5]

```
NetworkSpawnOp SpawnAsync (
    NetworkObject prefab,
    Vector3? position = null,
    Quaternion? rotation = null,
    PlayerRef? inputAuthority = null,
    OnBeforeSpawned onBeforeSpawned = null,
    NetworkSpawnFlags flags = default,
    NetworkObjectSpawnDelegate onCompleted = null )
```

Attempts to network instantiate a [NetworkObject](#) using a [NetworkObject](#) prefab. Note: position and rotation values are only used locally for the instantiation of the object, and are not inherently networked. Use [NetworkTransform](#) (or any custom class derived from [NetworkTRSP](#)) to replicate the initial transform state.

Parameters

<i>prefab</i>	Prefab used to spawn the NetworkObject
<i>position</i>	Spawn Position
<i>rotation</i>	Spawn Rotation
<i>inputAuthority</i>	Player Input Authority
<i>onBeforeSpawned</i>	OnBeforeSpawned reference
<i>flags</i>	Spawn flags

,

Parameters

<i>onCompleted</i>	A callback to fire once the spawn is done.
--------------------	--

6.250.3.97 SpawnAsync() [3/5]

```
NetworkSpawnOp SpawnAsync (
    NetworkObjectGuid prefabGuid,
    Vector3? position = null,
    Quaternion? rotation = null,
    PlayerRef? inputAuthority = null,
    OnBeforeSpawned onBeforeSpawned = null,
    NetworkSpawnFlags flags = default,
    NetworkObjectSpawnDelegate onCompleted = null )
```

Attempts to network instantiate a [NetworkObject](#) using a [NetworkObjectGuid](#) Note: position and rotation values are only used locally for the instantiation of the object, and are not inherently networked. Use [NetworkTransform](#) (or any custom class derived from [NetworkTRSP](#)) to replicate the initial transform state.

Parameters

<i>prefabGuid</i>	Object Guid used to spawn the NetworkObject
<i>position</i>	Spawn Position
<i>rotation</i>	Spawn Rotation

Parameters

<i>inputAuthority</i>	Player Input Authority
<i>onBeforeSpawned</i>	OnBeforeSpawned reference
<i>flags</i>	Spawn flags

,

Parameters

<i>onCompleted</i>	A callback to fire once the spawn is done.
--------------------	--

6.250.3.98 SpawnAsync() [4/5]

```
NetworkSpawnOp SpawnAsync (
    NetworkPrefabId typeId,
    Vector3? position = null,
    Quaternion? rotation = null,
    PlayerRef? inputAuthority = null,
    OnBeforeSpawned onBeforeSpawned = null,
    NetworkSpawnFlags flags = default,
    NetworkObjectSpawnDelegate onCompleted = null )
```

Attempts to network instantiate a [NetworkObject](#) using a [NetworkPrefabId](#) Note: position and rotation values are only used locally for the instantiation of the object, and are not inherently networked. Use [NetworkTransform](#) (or any custom class derived from [NetworkTRSP](#)) to replicate the initial transform state.

Parameters

<i>typeId</i>	Prefab ID used to spawn the NetworkObject
<i>position</i>	Spawn Position
<i>rotation</i>	Spawn Rotation
<i>inputAuthority</i>	Player Input Authority
<i>onBeforeSpawned</i>	OnBeforeSpawned reference
<i>flags</i>	Spawn flags

,

Parameters

<i>onCompleted</i>	A callback to fire once the spawn is done.
--------------------	--

6.250.3.99 SpawnAsync() [5/5]

```
NetworkSpawnOp SpawnAsync (
```

```

NetworkPrefabRef prefabRef,
Vector3? position = null,
Quaternion? rotation = null,
PlayerRef? inputAuthority = null,
OnBeforeSpawned onBeforeSpawned = null,
NetworkSpawnFlags flags = default,
NetworkObjectSpawnDelegate onCompleted = null )

```

Attempts to network instantiate a [NetworkObject](#) using a [NetworkPrefabRef](#). Note: position and rotation values are only used locally for the instantiation of the object, and are not inherently networked. Use [NetworkTransform](#) (or any custom class derived from [NetworkTRSP](#)) to replicate the initial transform state.

Parameters

<i>prefabRef</i>	Prefab Ref used to spawn the NetworkObject
<i>position</i>	Spawn Position
<i>rotation</i>	Spawn Rotation
<i>inputAuthority</i>	Player Input Authority
<i>onBeforeSpawned</i>	OnBeforeSpawned reference
<i>flags</i>	Spawn flags

Parameters

<i>onCompleted</i>	A callback to fire once the spawn is done.
--------------------	--

6.250.3.100 SpawnAsync< T >()

```

NetworkSpawnOp SpawnAsync< T > (
    T prefab,
    Vector3? position = null,
    Quaternion? rotation = null,
    PlayerRef? inputAuthority = null,
    OnBeforeSpawned onBeforeSpawned = null,
    NetworkSpawnFlags flags = default,
    NetworkObjectSpawnDelegate onCompleted = null )

```

Attempts to network instantiate a [NetworkObject](#) using a Component type that is part of a [NetworkObject](#)

Template Parameters

<i>T</i>	Must be a Type derived from SimulationBehaviour
----------	---

Parameters

<i>prefab</i>	SimulationBehaviour used to spawn the NetworkObject
---------------	---

Returns

T reference, or null if it was not able to spawn the object

Parameters

<i>onCompleted</i>	A callback to fire once the spawn is done.
<i>position</i>	Spawn Position
<i>rotation</i>	Spawn Rotation
<i>inputAuthority</i>	Player Input Authority
<i>onBeforeSpawned</i>	OnBeforeSpawned reference
<i>flags</i>	Spawn flags

Type Constraints

T : *SimulationBehaviour*

6.250.3.101 StartGame()

```
Task<StartGameResult> StartGame (
    StartGameArgs args )
```

Starts the local [Fusion](#) Runner and takes care of all major setup necessary

More about matchmaking: <https://doc.photonengine.com/en-us/fusion/current/manual/matchmaking>

Parameters

<i>args</i>	Custom arguments used to setup the Fusion Simulation
-------------	--

Returns

Task that can be awaited to chain actions

6.250.3.102 TryFindBehaviour()

```
bool TryFindBehaviour (
    NetworkBehaviourId behaviourId,
    out NetworkBehaviour behaviour )
```

Get the [NetworkBehaviour](#) instance for this [NetworkRunner](#) from a [NetworkBehaviourId](#).

Parameters

<i>behaviour</i> ↔ <i>Id</i>	NetworkBehaviourId to look forward
<i>behaviour</i>	NetworkBehaviour reference, if found

Returns

True if object was found.

6.250.3.103 TryFindBehaviour< T >()

```
bool TryFindBehaviour< T > (
    NetworkBehaviourId id,
    out T behaviour )
```

Try to find a [NetworkBehaviour](#) with the provided [NetworkBehaviourId](#).

Parameters

<i>id</i>	The NetworkBehaviourId to search for
<i>behaviour</i>	The behaviour found

Template Parameters

<i>T</i>	A NetworkBehaviour type
----------	---

Returns

Returns true if the behaviour was found and it is alive. False otherwise

Type Constraints

***T* : [NetworkBehaviour](#)**

6.250.3.104 TryFindObject()

```
bool TryFindObject (
    NetworkId objectId,
    out NetworkObject networkObject )
```

Get the [NetworkObject](#) instance for this [NetworkRunner](#) from a [NetworkId](#).

Parameters

<i>objectId</i>	Object NetworkID to look forward
<i>networkObject</i>	NetworkObject reference, if found

Returns

True if object was found.

6.250.3.105 TryGetBehaviourStatistics()

```
bool TryGetBehaviourStatistics (
    Type behaviourType,
    out BehaviourStatisticsSnapshot behaviourStatisticsSnapshot )
```

Tries to get the statistics snapshot for a specified behaviour type.

Parameters

<i>behaviourType</i>	The type of the behaviour for which to get the statistics snapshot.
<i>behaviourStatisticsSnapshot</i>	When this method returns, contains the statistics snapshot for the specified behaviour type, if found; otherwise, the default value.

Returns

`true` if the statistics snapshot for the specified behaviour type is found; otherwise, `false`.

6.250.3.106 TryGetFusionStatistics()

```
bool TryGetFusionStatistics (
    out FusionStatisticsManager statisticsManager )
```

Tries to get the FusionStatisticsManager from the simulation.

Parameters

<i>statisticsManager</i>	The FusionStatisticsManager returned by the method
--------------------------	--

Returns

True if the FusionStatisticsManager is successfully retrieved, otherwise false

6.250.3.107 TryGetInputForPlayer< T >()

```
bool TryGetInputForPlayer< T > (
    PlayerRef player,
    out T input )
```

Outputs the [NetworkInput](#) from player, translated to the indicated [INetworkInput](#).

Type Constraints

T* : *unmanaged

T* : *INetworkInput

6.250.3.108 TryGetNetworkedBehaviourFromNetworkedObjectRef< T >()

```
T TryGetNetworkedBehaviourFromNetworkedObjectRef< T > (
    NetworkId networkId )
```

Tries to return the first instance of T found on the root of a [NetworkObject](#).

Template Parameters

<i>T</i>	The type of the component to search for
----------	---

Parameters

<i>networkId</i>	NetworkId of the NetworkObject to search for
------------------	--

Returns

Returns the found component. Null if the [NetworkObject](#) cannot be found, or if T cannot be found on the [GameObject](#).

Type Constraints

T* : *NetworkBehaviour

6.250.3.109 TryGetNetworkedBehaviourId()

```
NetworkBehaviourId TryGetNetworkedBehaviourId (
    NetworkBehaviour behaviour )
```

Tries to return a [NetworkBehaviourId](#) for the [NetworkBehaviour](#) provided.

Parameters

<i>behaviour</i>	NetworkBehaviour to get the NetworkBehaviourId from
------------------	---

Returns

Returns a [NetworkBehaviourId](#) to the provided behaviour. Returns default if the behaviour is not alive or the [NetworkObject](#) that has this behaviour is not valid.

6.250.3.110 TryGetObjectRefFromNetworkedBehaviour()

```
NetworkId TryGetObjectRefFromNetworkedBehaviour (
    NetworkBehaviour behaviour )
```

Tries to return the behaviour [NetworkId](#).

Parameters

<i>behaviour</i>	NetworkBehaviour to get the NetworkId from
------------------	--

Returns

Returns the [NetworkId](#) of the provided behaviour. Returns default if the behaviour is not alive or the [NetworkObject](#) that has this behaviour is not valid.

6.250.3.111 TryGetPhysicsInfo()

```
bool TryGetPhysicsInfo (
    out NetworkPhysicsInfo info )
```

Try to get the physics info.

Parameters

<i>info</i>	Network physics info
-------------	----------------------

Returns

True if the physics info exists, otherwise false.

6.250.3.112 TryGetPlayerObject()

```
bool TryGetPlayerObject (
    PlayerRef player,
    out NetworkObject networkObject )
```

Try to gets the [NetworkObject](#) associated with a specific player

Parameters

<i>player</i>	PlayerRef to get the network object
<i>networkObject</i>	Network object if one is associated with the player

Returns

Signals if it was able to get a [NetworkObject](#) for the player provided

6.250.3.113 TryGetSceneInfo()

```
bool TryGetSceneInfo (
    out NetworkSceneInfo sceneInfo )
```

Tries to get the [NetworkSceneInfo](#) of this [NetworkRunner](#).

Parameters

<i>sceneInfo</i>	The result NetworkSceneInfo
------------------	---

Returns

Returns true if it was able to get the scene info

6.250.3.114 TrySetPhysicsInfo()

```
bool TrySetPhysicsInfo (
    NetworkPhysicsInfo info )
```

Try to set the physics info.

Parameters

<i>info</i>	Network physics info
-------------	----------------------

Returns

True if the physics info was set, otherwise false.

Exceptions

<i>InvalidOperationException</i>	Thrown if the runner does not have the scene authority.
----------------------------------	---

6.250.3.115 TrySpawn() [1/5]

```
NetworkSpawnStatus TrySpawn (
    GameObject prefab,
    out NetworkObject obj,
    Vector3? position = null,
    Quaternion? rotation = null,
    PlayerRef? inputAuthority = null,
    OnBeforeSpawned onBeforeSpawned = null,
    NetworkSpawnFlags flags = default )
```

Attempts to network instantiate a [NetworkObject](#) using a [GameObject](#). The supplied [GameObject](#) must have a [NetworkObject](#) component.

Parameters

<i>prefab</i>	A GameObject with a NetworkObject
<i>position</i>	Spawn Position
<i>rotation</i>	Spawn Rotation
<i>inputAuthority</i>	Player Input Authority
<i>onBeforeSpawned</i>	OnBeforeSpawned reference
<i>flags</i>	Spawn flags
<i>obj</i>	Spawned NetworkObject reference

Returns

[NetworkSpawnStatus](#) reference, or null if it was not able to spawn the object

6.250.3.116 TrySpawn() [2/5]

```
NetworkSpawnStatus TrySpawn (
    NetworkObject prefab,
    out NetworkObject obj,
    Vector3? position = null,
    Quaternion? rotation = null,
    PlayerRef? inputAuthority = null,
    OnBeforeSpawned onBeforeSpawned = null,
    NetworkSpawnFlags flags = default )
```

Attempts to network instantiate a [NetworkObject](#) using a [NetworkObject](#) prefab. Note: position and rotation values are only used locally for the instantiation of the object, and are not inherently networked. Use [NetworkTransform](#) (or any custom class derived from [NetworkTRSP](#)) to replicate the initial transform state.

Parameters

<i>prefab</i>	Prefab used to spawn the NetworkObject
<i>position</i>	Spawn Position
<i>rotation</i>	Spawn Rotation
<i>inputAuthority</i>	Player Input Authority
<i>onBeforeSpawned</i>	OnBeforeSpawned reference
<i>flags</i>	Spawn flags
<i>obj</i>	Spawned NetworkObject reference

Returns

[NetworkSpawnStatus](#) reference, or null if it was not able to spawn the object

6.250.3.117 TrySpawn() [3/5]

```
NetworkSpawnStatus TrySpawn (
    NetworkObjectGuid prefabGuid,
    out NetworkObject obj,
    Vector3? position = null,
    Quaternion? rotation = null,
    PlayerRef? inputAuthority = null,
    OnBeforeSpawned onBeforeSpawned = null,
    NetworkSpawnFlags flags = default )
```

Attempts to network instantiate a [NetworkObject](#) using a [NetworkObjectGuid](#) Note: position and rotation values are only used locally for the instantiation of the object, and are not inherently networked. Use [NetworkTransform](#) (or any custom class derived from [NetworkTRSP](#)) to replicate the initial transform state.

Parameters

<i>prefabGuid</i>	Object Guid used to spawn the NetworkObject
<i>position</i>	Spawn Position
<i>rotation</i>	Spawn Rotation
<i>inputAuthority</i>	Player Input Authority
<i>onBeforeSpawned</i>	OnBeforeSpawned reference
<i>flags</i>	Spawn flags
<i>obj</i>	Spawned NetworkObject reference

Returns

[NetworkSpawnStatus](#) reference, or null if it was not able to spawn the object

6.250.3.118 TrySpawn() [4/5]

```
NetworkSpawnStatus TrySpawn (
    NetworkPrefabId typeId,
    out NetworkObject obj,
    Vector3? position = null,
    Quaternion? rotation = null,
    PlayerRef? inputAuthority = null,
    OnBeforeSpawned onBeforeSpawned = null,
    NetworkSpawnFlags flags = default )
```

Attempts to network instantiate a [NetworkObject](#) using a [NetworkPrefabId](#) Note: position and rotation values are only used locally for the instantiation of the object, and are not inherently networked. Use [NetworkTransform](#) (or any custom class derived from [NetworkTRSP](#)) to replicate the initial transform state.

Parameters

<i>typeId</i>	Prefab ID used to spawn the NetworkObject
<i>position</i>	Spawn Position
<i>rotation</i>	Spawn Rotation
<i>inputAuthority</i>	Player Input Authority
<i>onBeforeSpawned</i>	OnBeforeSpawned reference
<i>flags</i>	Spawn flags
<i>obj</i>	Spawned NetworkObject reference

Returns

[NetworkSpawnStatus](#) reference, or null if it was not able to spawn the object

6.250.3.119 TrySpawn() [5/5]

```
NetworkSpawnStatus TrySpawn (
    NetworkPrefabRef prefabRef,
    out NetworkObject obj,
    Vector3? position = null,
    Quaternion? rotation = null,
    PlayerRef? inputAuthority = null,
    OnBeforeSpawned onBeforeSpawned = null,
    NetworkSpawnFlags flags = default )
```

Attempts to network instantiate a [NetworkObject](#) using a [NetworkPrefabRef](#). Note: position and rotation values are only used locally for the instantiation of the object, and are not inherently networked. Use [NetworkTransform](#) (or any custom class derived from [NetworkTRSP](#)) to replicate the initial transform state.

Parameters

<i>prefabRef</i>	Prefab Ref used to spawn the NetworkObject
<i>position</i>	Spawn Position
<i>rotation</i>	Spawn Rotation
<i>inputAuthority</i>	Player Input Authority
<i>onBeforeSpawned</i>	OnBeforeSpawned reference
<i>flags</i>	Spawn flags
<i>obj</i>	Spawned NetworkObject reference

Returns

[NetworkSpawnStatus](#) reference, or null if it was not able to spawn the object

6.250.3.120 TrySpawn< T >()

```
NetworkSpawnStatus TrySpawn< T > (
    T prefab,
    out T obj,
    Vector3? position = null,
    Quaternion? rotation = null,
    PlayerRef? inputAuthority = null,
    OnBeforeSpawned onBeforeSpawned = null,
    NetworkSpawnFlags flags = default )
```

Attempts to network instantiate a [NetworkObject](#) using a Component type that is part of a [NetworkObject](#)

Template Parameters

<i>T</i>	Must be a Type derived from SimulationBehaviour
----------	---

Parameters

<i>prefab</i>	SimulationBehaviour used to spawn the NetworkObject
---------------	---

Returns

T reference, or null if it was not able to spawn the object

Parameters

<i>obj</i>	Spawned NetworkObject reference
<i>position</i>	Spawn Position
<i>rotation</i>	Spawn Rotation
<i>inputAuthority</i>	Player Input Authority
<i>onBeforeSpawned</i>	OnBeforeSpawned reference
<i>flags</i>	Spawn flags

Type Constraints

***T* : [SimulationBehaviour](#)**

6.250.3.121 UnloadScene()

```
NetworkSceneAsyncOp UnloadScene (
    string sceneName )
```

Unloads a scene

Parameters

<code>sceneName</code>	Name of the scene to unload
------------------------	-----------------------------

Returns

Scene Unload operation

6.250.3.122 UpdateInternal()

```
void UpdateInternal (
    double dt )
```

This method is meant to be called by [INetworkRunnerUpdater](#).

6.250.4 Property Documentation

6.250.4.1 ActivePlayers

```
IEnumerable<PlayerRef> ActivePlayers [get]
```

Returns the collection of [PlayerRef](#) objects for this [NetworkRunner](#)'s [Fusion.Simulation](#).

6.250.4.2 AuthenticationValues

```
AuthenticationValues AuthenticationValues [get]
```

[AuthenticationValues](#) used by this Runner to Authenticate the local peer.

6.250.4.3 BuildType

```
BuildTypes BuildType [static], [get]
```

Get Fusion.Runtime.dll build type.

6.250.4.4 CanSpawn

```
bool CanSpawn [get]
```

Signal if the Network Runner can spawn a [NetworkObject](#)

6.250.4.5 Config

```
NetworkProjectConfig Config [get]
```

Returns the [NetworkProjectConfig](#) reference.

6.250.4.6 CurrentConnectionType

```
ConnectionType CurrentConnectionType [get]
```

Check the current Connection Type with the Remote Server

6.250.4.7 DeltaTime

```
float DeltaTime [get]
```

Returns the fixed tick time interval. Derived from the [SimulationRuntimeConfig.TickRate](#).

6.250.4.8 GameMode

```
GameMode GameMode [get]
```

Current Game Mode active on the [Fusion Simulation](#)

6.250.4.9 Instances

```
IReadOnlyList<NetworkRunner> Instances [static], [get]
```

A list of all [NetworkRunners](#).

6.250.4.10 IsClient

```
bool IsClient [get]
```

Returns if this [Fusion.Simulation](#) represents a Client connection.

6.250.4.11 IsCloudReady

```
bool IsCloudReady [get]
```

Signal if the Local Peer is connected to Photon Cloud and is able to Create/Join Room but also receive Lobby Updates

6.250.4.12 IsConnectedToServer

```
bool IsConnectedToServer [get]
```

Returns if this Client is currently connected to a Remote Server

6.250.4.13 IsFirstTick

```
bool IsFirstTick [get]
```

If this is the first tick that executes this update or re-simulation

6.250.4.14 IsForward

```
bool IsForward [get]
```

If this is not a re-simulation but a new forward tick

6.250.4.15 IsLastTick

```
bool IsLastTick [get]
```

If this is the last tick that is being executed this update

6.250.4.16 IsPlayer

```
bool IsPlayer [get]
```

Returns true if this runner represents a Client or Host. Dedicated servers have no local player and will return false.

6.250.4.17 IsResimulation

```
bool IsResimulation [get]
```

If we are currently executing a client side prediction re-simulation.

6.250.4.18 IsResume

```
bool IsResume [get]
```

if this instance is a resume (host migration)

6.250.4.19 IsRunning

```
bool IsRunning [get]
```

Returns if this [Fusion.Simulation](#) is valid and running.

6.250.4.20 IsSceneAuthority

```
bool IsSceneAuthority [get]
```

Is this runner responsible for scene management.

6.250.4.21 IsSceneManagerBusy

```
bool? IsSceneManagerBusy [get]
```

Signals if the [INetworkSceneManager](#) instance assigned to this [NetworkRunner](#) is busy with any scene loading operation.

6.250.4.22 IsServer

```
bool IsServer [get]
```

Returns if this [Fusion.Simulation](#) represents a Server connection.

6.250.4.23 IsSharedModeMasterClient

```
bool IsSharedModeMasterClient [get]
```

Signal if the Local Peer is in a Room and is the Room Master Client

6.250.4.24 IsShutdown

```
bool IsShutdown [get]
```

If the runner is shutdown

6.250.4.25 IsSimulationUpdating

```
bool IsSimulationUpdating [get]
```

Is the runner updating the simulation.

6.250.4.26 IsSinglePlayer

```
bool IsSinglePlayer [get]
```

Returns true if this runner was started as single player (Started as [SimulationModes.Host](#) with [SimulationConfig.PlayerCount](#) = 1).

6.250.4.27 IsStarting

```
bool IsStarting [get]
```

If the runner is pending to start

6.250.4.28 LagCompensation

`HitboxManager LagCompensation [get]`

Returns the global instance of a lag compensation buffer [Fusion.HitboxManager](#).

6.250.4.29 LatestServerTick

`Tick LatestServerTick [get]`

Get the latest confirmed tick of the server we are aware of

6.250.4.30 LobbyInfo

`LobbyInfo LobbyInfo = new LobbyInfo() [get]`

Signal if the local peer is already inside a Lobby

6.250.4.31 LocalAlpha

`float LocalAlpha [get]`

Get the local time alpha value

6.250.4.32 LocalPlayer

`PlayerRef LocalPlayer [get]`

Returns a [PlayerRef](#) for the local simulation. For a dedicated server [PlayerRef.IsRealPlayer](#) will equal false. [PlayerRef](#)s are assigned in order from 0 to `MaxPlayers-1` and are re-used as players join and leave. The only caveat is that the server player (if one exists), always gets the last index no matter how many clients are connected.

6.250.4.33 LocalRenderTime

`float??? LocalRenderTime [get]`

The current time (current `State.Time` + [Simulation.DeltaTime](#)) for predicted objects (objects in the local time frame). Use as an equivalent to Unity's `Time.time`. Time is relative to [Tick 0](#) (which represents `Time 0f`).

6.250.4.34 Mode

`SimulationModes Mode [get]`

Returns the [SimulationModes](#) flags for The type of network peer the associated [Fusion.Simulation](#) represents.

6.250.4.35 NATType

`NATType NATType [get]`

Exposes the current NAT Type from the local Peer

6.250.4.36 ObjectProvider

`INetworkObjectProvider ObjectProvider [get]`

Returns the [INetworkObjectProvider](#) instance.

6.250.4.37 Prefabs

`NetworkPrefabTable Prefabs [get]`

Reference to the [NetworkPrefabTable](#).

6.250.4.38 ProvideInput

`bool ProvideInput [get], [set]`

Indicates if this [NetworkRunner](#) is collecting [PlayerRef INetworkInput](#).

6.250.4.39 RemoteRenderTime

`float RemoteRenderTime [get]`

The current time (current `State.Time` + [Simulation.DeltaTime](#)) for non-predicted objects (objects in a remote time frame). Use as an equivalent to Unity's `Time.time`. Time is relative to [Tick 0](#) (which represents Time 0f).

6.250.4.40 SceneManager

```
INetworkSceneManager SceneManager [get]
```

Returns the [INetworkSceneManager](#) instance.

6.250.4.41 SessionInfo

```
SessionInfo SessionInfo = new SessionInfo() [get]
```

Stores information about the current running session

6.250.4.42 SimulationTime

```
float SimulationTime [get]
```

The time the current State represents (the most recent FixedUpdateNetwork simulation). Use as an equivalent to Unity's Time.fixedTime. Time is relative to [Tick 0](#) (which represents Time 0f).

6.250.4.43 SimulationUnityScene

```
Scene??? SimulationUnityScene [get]
```

The main scene of the [NetworkRunner](#) or default if not running.

6.250.4.44 Stage

```
SimulationStages Stage [get]
```

Returns the current [SimulationStages](#) stage of this [Fusion.Simulation](#).

6.250.4.45 State

```
States State [get]
```

The current state of the runner, if it's Starting, Running, Shutdown

6.250.4.46 Tick

```
Tick??? Tick [get]
```

The tick associated with the current state of networked objects, or the current simulation tick being processed (when evaluated during FixedUpdateNetwork).

6.250.4.47 TickRate

```
int TickRate [get]
```

6.250.4.48 TicksExecuted

```
int TicksExecuted [get]
```

Returns how many ticks we executed last update.

6.250.4.49 Topology

```
Topologies Topology [get]
```

The current topology used

6.250.4.50 UserId

```
string UserId [get]
```

Photon Client UserID

Returns null if Peer is not connected to Photon Cloud

6.250.5 Event Documentation

6.250.5.1 ObjectAcquired

```
ObjectDelegate ObjectAcquired
```

Event for object acquired

6.251 NetworkRunnerCallbackArgs Class Reference

Stores data types used on the [INetworkRunnerCallbacks](#) interface

Classes

- class [ConnectRequest](#)
Data holder of a Connection Request from a remote client

6.251.1 Detailed Description

Stores data types used on the [INetworkRunnerCallbacks](#) interface

6.252 NetworkRunnerCallbackArgs.ConnectRequest Class Reference

Data holder of a Connection Request from a remote client

Public Member Functions

- void [Accept](#) ()
Accepts the Request
- void [Refuse](#) ()
Refuses the Request
- void [Waiting](#) ()
Refuses the Request

Properties

- [NetAddress RemoteAddress](#) [get, set]
Address of the remote client

6.252.1 Detailed Description

Data holder of a Connection Request from a remote client

6.252.2 Member Function Documentation

6.252.2.1 Accept()

```
void Accept ( )
```

Accepts the Request

6.252.2.2 Refuse()

```
void Refuse ( )
```

Refuses the Request

6.252.2.3 Waiting()

```
void Waiting ( )
```

Refuses the Request

6.252.3 Property Documentation

6.252.3.1 RemoteAddress

```
NetAddress RemoteAddress [get], [set]
```

Address of the remote client

6.253 NetworkRunnerUpdaterDefault Class Reference

Default implementation of [INetworkRunnerUpdater](#) that uses the Unity PlayerLoop.

Inherits [INetworkRunnerUpdater](#).

Classes

- struct [NetworkRunnerRender](#)
Used to invoke [NetworkRunner.RenderInternal](#) in the PlayerLoop.
- struct [NetworkRunnerUpdate](#)
Used to invoke [NetworkRunner.UpdateInternal\(double\)](#) in the PlayerLoop.

Static Public Member Functions

- static bool [RegisterInPlayerLoop](#) ([NetworkRunnerUpdaterDefaultInvokeSettings](#) updateSettings, [NetworkRunnerUpdaterDefaultInvokeSettings](#) renderSettings)
Registers in the PlayerLoop.
- static bool [UnregisterFromPlayerLoop](#) ()
Unregisters from the PlayerLoop.

Public Attributes

- [NetworkRunnerUpdaterDefaultInvokeSettings](#) [RenderSettings](#)
Default settings for the [NetworkRunner](#) Render Loop.
- [NetworkRunnerUpdaterDefaultInvokeSettings](#) [UpdateSettings](#)
Default settings for the [NetworkRunner](#) Update Loop.

Additional Inherited Members

6.253.1 Detailed Description

Default implementation of [INetworkRunnerUpdater](#) that uses the Unity PlayerLoop.

6.253.2 Member Function Documentation

6.253.2.1 RegisterInPlayerLoop()

```
static bool RegisterInPlayerLoop (
    NetworkRunnerUpdaterDefaultInvokeSettings updateSettings,
    NetworkRunnerUpdaterDefaultInvokeSettings renderSettings ) [static]
```

Registers in the PlayerLoop.

Parameters

<i>updateSettings</i>	Update settings.
<i>renderSettings</i>	Render settings.

Returns

True if registered, false if already registered with the same settings.

6.253.2 UnregisterFromPlayerLoop()

```
static bool UnregisterFromPlayerLoop ( ) [static]
```

Unregisters from the PlayerLoop.

Returns

True if unregistered, false if not registered.

6.253.3 Member Data Documentation

6.253.3.1 RenderSettings

[NetworkRunnerUpdaterDefaultInvokeSettings](#) RenderSettings

Initial value:

```
= new NetworkRunnerUpdaterDefaultInvokeSettings {  
    ReferencePlayerLoopSystem = typeof(Update.ScriptRunBehaviourUpdate),  
    AddMode                   = UnityPlayerLoopSystemAddMode.After  
}
```

Default settings for the [NetworkRunner](#) Render Loop.

6.253.3.2 UpdateSettings

[NetworkRunnerUpdaterDefaultInvokeSettings](#) UpdateSettings

Initial value:

```
= new NetworkRunnerUpdaterDefaultInvokeSettings {  
    ReferencePlayerLoopSystem = typeof(Update.ScriptRunBehaviourUpdate),  
    AddMode                   = UnityPlayerLoopSystemAddMode.Before  
}
```

Default settings for the [NetworkRunner](#) Update Loop.

6.254 NetworkRunnerUpdaterDefault.NetworkRunnerRender Struct Reference

Used to invoke [NetworkRunner.RenderInternal](#) in the PlayerLoop.

6.254.1 Detailed Description

Used to invoke [NetworkRunner.RenderInternal](#) in the PlayerLoop.

6.255 NetworkRunnerUpdaterDefault.NetworkRunnerUpdate Struct Reference

Used to invoke [NetworkRunner.UpdateInternal\(double\)](#) in the PlayerLoop.

6.255.1 Detailed Description

Used to invoke [NetworkRunner.UpdateInternal\(double\)](#) in the PlayerLoop.

6.256 NetworkRunnerUpdaterDefaultInvokeSettings Struct Reference

Settings for the [NetworkRunnerUpdaterDefault](#).

Inherits [IEquatable< NetworkRunnerUpdaterDefaultInvokeSettings >](#).

Public Member Functions

- bool [Equals](#) ([NetworkRunnerUpdaterDefaultInvokeSettings](#) other)
Checks if the settings are equal.
- override bool [Equals](#) (object obj)
Checks if the settings are equal.
- override int [GetHashCode](#) ()
Gets the hash code of the settings.
- override string [ToString](#) ()
Returns a string representation of the settings.

Static Public Member Functions

- static bool [operator!=](#) ([NetworkRunnerUpdaterDefaultInvokeSettings](#) left, [NetworkRunnerUpdaterDefaultInvokeSettings](#) right)
Checks if the settings are not equal.
- static bool [operator==](#) ([NetworkRunnerUpdaterDefaultInvokeSettings](#) left, [NetworkRunnerUpdaterDefaultInvokeSettings](#) right)
Checks if the settings are equal.

Public Attributes

- [UnityPlayerLoopSystemAddMode](#) AddMode
Add mode for the PlayerLoopSystem.
- Type [ReferencePlayerLoopSystem](#)
Reference to the PlayerLoopSystem to add the [NetworkRunner](#) to.

6.256.1 Detailed Description

Settings for the [NetworkRunnerUpdaterDefault](#).

6.256.2 Member Function Documentation

6.256.2.1 Equals() [1/2]

```
bool Equals (  
    NetworkRunnerUpdaterDefaultInvokeSettings other )
```

Checks if the settings are equal.

Parameters

<i>other</i>	Settings to check for equality.
--------------	---------------------------------

Returns

True if equal, false otherwise.

6.256.2.2 Equals() [2/2]

```
override bool Equals (  
    object obj )
```

Checks if the settings are equal.

Parameters

<i>obj</i>	Settings to check for equality.
------------	---------------------------------

Returns

True if equal, false otherwise.

6.256.2.3 GetHashCode()

```
override int GetHashCode ( )
```

Gets the hash code of the settings.

Returns

Hash code.

6.256.2.4 operator"!=()

```
static bool operator!= (
    NetworkRunnerUpdaterDefaultInvokeSettings left,
    NetworkRunnerUpdaterDefaultInvokeSettings right ) [static]
```

Checks if the settings are not equal.

Parameters

<i>left</i>	Settings to check for equality.
<i>right</i>	Settings to check for equality.

Returns

True if not equal, false otherwise.

6.256.2.5 operator==(())

```
static bool operator==(
    NetworkRunnerUpdaterDefaultInvokeSettings left,
    NetworkRunnerUpdaterDefaultInvokeSettings right ) [static]
```

Checks if the settings are equal.

Parameters

<i>left</i>	Settings to check for equality.
<i>right</i>	Settings to check for equality.

Returns

True if equal, false otherwise.

6.256.2.6 ToString()

```
override string ToString ( )
```

Returns a string representation of the settings.

Returns

String representation.

6.256.3 Member Data Documentation

6.256.3.1 AddMode

`UnityPlayerLoopSystemAddMode` AddMode

Add mode for the PlayerLoopSystem.

6.256.3.2 ReferencePlayerLoopSystem

Type `ReferencePlayerLoopSystem`

Reference to the PlayerLoopSystem to add the [NetworkRunner](#) to.

6.257 NetworkSceneAsyncOp Struct Reference

A wrapper for async scene operations.

Inherits `IEnumerator`.

Classes

- struct [Awaiter](#)
Awaiter for NetworkSceneAsyncOp

Public Member Functions

- void [AddOnCompleted](#) (Action< [NetworkSceneAsyncOp](#) > action)
Adds a callback to be called when the operation is completed
- [Awaiter](#) [GetAwaiter](#) ()
Gets the awaiter for the operation
- bool `IEnumerator`. **MoveNext** ()
- void `IEnumerator`. **Reset** ()

Static Public Member Functions

- static [NetworkSceneAsyncOp](#) [FromAsyncOperation](#) ([SceneRef](#) sceneRef, `UnityEngine.AsyncOperation` asyncOp)
Creates a [NetworkSceneAsyncOp](#) from a `UnityEngine.AsyncOperation`
- static [NetworkSceneAsyncOp](#) [FromCompleted](#) ([SceneRef](#) sceneRef)
Creates a completed [NetworkSceneAsyncOp](#)
- static [NetworkSceneAsyncOp](#) [FromCoroutine](#) ([SceneRef](#) sceneRef, [ICoroutine](#) coroutine)
Creates a [NetworkSceneAsyncOp](#) from a `ICoroutine`
- static [NetworkSceneAsyncOp](#) [FromError](#) ([SceneRef](#) sceneRef, Exception error)
Creates a [NetworkSceneAsyncOp](#) from a `Exception`
- static [NetworkSceneAsyncOp](#) [FromTask](#) ([SceneRef](#) sceneRef, `Task` task)
Creates a [NetworkSceneAsyncOp](#) from a `Task`

Public Attributes

- readonly [SceneRef](#) **SceneRef**
The scene reference of the operation

Properties

- object IEnumerator. **Current** [get]
- Exception? [Error](#) [get]
Attached error to the operation
- bool [IsDone](#) [get]
Signals if the operation is done
- bool [IsValid](#) [get]
Signals if the operation is valid

6.257.1 Detailed Description

A wrapper for async scene operations.

6.257.2 Member Function Documentation

6.257.2.1 AddOnCompleted()

```
void AddOnCompleted (
    Action< NetworkSceneAsyncOp > action )
```

Adds a callback to be called when the operation is completed

Parameters

<i>action</i>	The callback to be called
---------------	---------------------------

6.257.2.2 FromAsyncOperation()

```
static NetworkSceneAsyncOp FromAsyncOperation (
    SceneRef sceneRef,
    UnityEngine.AsyncOperation asyncOp ) [static]
```

Creates a [NetworkSceneAsyncOp](#) from a UnityEngine.AsyncOperation

Parameters

<i>sceneRef</i>	Scene reference
<i>asyncOp</i>	Async operation reference

Returns

Returns a [NetworkSceneAsyncOp](#) instance

Exceptions

<i>ArgumentNullException</i>	Thrown if <i>asyncOp</i> is null
------------------------------	----------------------------------

6.257.2.3 FromCompleted()

```
static NetworkSceneAsyncOp FromCompleted (
    SceneRef sceneRef ) [static]
```

Creates a completed [NetworkSceneAsyncOp](#)

Parameters

<i>sceneRef</i>	Scene reference
-----------------	-----------------

Returns

Returns a [NetworkSceneAsyncOp](#) instance

6.257.2.4 FromCoroutine()

```
static NetworkSceneAsyncOp FromCoroutine (
    SceneRef sceneRef,
    ICoroutine coroutine ) [static]
```

Creates a [NetworkSceneAsyncOp](#) from a [ICoroutine](#)

Parameters

<i>sceneRef</i>	Scene reference
<i>coroutine</i>	Coroutine reference

Returns

Returns a [NetworkSceneAsyncOp](#) instance

Exceptions

<i>ArgumentNullException</i>	Thrown if <i>coroutine</i> is null
------------------------------	------------------------------------

6.257.2.5 FromError()

```
static NetworkSceneAsyncOp FromError (
    SceneRef sceneRef,
    Exception error ) [static]
```

Creates a [NetworkSceneAsyncOp](#) from a Exception

Parameters

<i>sceneRef</i>	Scene reference
<i>error</i>	Exception reference

Returns

Returns a [NetworkSceneAsyncOp](#) instance

Exceptions

<i>ArgumentNullException</i>	Thrown if <i>error</i> is null
------------------------------	--------------------------------

6.257.2.6 FromTask()

```
static NetworkSceneAsyncOp FromTask (
    SceneRef sceneRef,
    Task task ) [static]
```

Creates a [NetworkSceneAsyncOp](#) from a Task

Parameters

<i>sceneRef</i>	Scene reference
<i>task</i>	Task reference

Returns

Returns a [NetworkSceneAsyncOp](#) instance

Exceptions

<i>ArgumentNullException</i>	Thrown if <i>task</i> is null
------------------------------	-------------------------------

6.257.2.7 GetAwaiter()

```
Awaiter GetAwaiter ( )
```

Gets the awaiter for the operation

6.257.3 Member Data Documentation

6.257.3.1 SceneRef

```
readonly SceneRef SceneRef
```

The scene reference of the operation

6.257.4 Property Documentation

6.257.4.1 Error

```
Exception? Error [get]
```

Attached error to the operation

6.257.4.2 IsDone

```
bool IsDone [get]
```

Signals if the operation is done

6.257.4.3 IsValid

```
bool IsValid [get]
```

Signals if the operation is valid

6.258 NetworkSceneAsyncOp.Awaiter Struct Reference

[Awaiter](#) for [NetworkSceneAsyncOp](#)

Inherits [INotifyCompletion](#).

Public Member Functions

- [Awaiter](#) (in [NetworkSceneAsyncOp](#) op)
Creates a new [Awaiter](#) instance
- void [GetResult](#) ()
Gets the result of the operation
- void [OnCompleted](#) (Action continuation)
Adds a callback to be called when the operation is completed

Public Attributes

- [NetworkSceneAsyncOp_op](#)

Properties

- bool [IsCompleted](#) [get]
Signals if the operation is completed

6.258.1 Detailed Description

[Awaiter](#) for [NetworkSceneAsyncOp](#)

6.258.2 Constructor & Destructor Documentation

6.258.2.1 Awaiter()

```
Awaiter (  
    in NetworkSceneAsyncOp op )
```

Creates a new [Awaiter](#) instance

Parameters

<i>op</i>	The operation to await
-----------	------------------------

6.258.3 Member Function Documentation

6.258.3.1 GetResult()

```
void GetResult ( )
```

Gets the result of the operation

6.258.3.2 OnCompleted()

```
void OnCompleted (
    Action continuation )
```

Adds a callback to be called when the operation is completed

Parameters

<i>continuation</i>	The callback to be called
---------------------	---------------------------

6.258.4 Property Documentation

6.258.4.1 IsCompleted

```
bool IsCompleted [get]
```

Signals if the operation is completed

6.259 NetworkSceneInfo Struct Reference

Can store up to 8 active scenes and allows for duplicates. Each write increases [Version](#) which can be used to generate unique scene objects ids for when a scene is supposed to be reloaded.

Inherits [INetworkStruct](#), and [IEquatable< NetworkSceneInfo >](#).

Public Member Functions

- int [AddSceneRef](#) ([SceneRef](#) sceneRef, LoadSceneMode loadSceneMode=LoadSceneMode.Single, Local↔PhysicsMode localPhysicsMode=LocalPhysicsMode.None, bool activeOnLoad=false)
Adds a scene to the list
- bool [Equals](#) ([NetworkSceneInfo](#) other)
Compares two [NetworkSceneInfo](#) for equality
- override bool [Equals](#) (object obj)
Compares two [NetworkSceneInfo](#) for equality
- override int [GetHashCode](#) ()
Get the hash code of the [NetworkSceneInfo](#)
- int [IndexOf](#) (([SceneRef](#) sceneRef, [NetworkLoadSceneParameters](#) sceneParams) scene)
- int [IndexOf](#) ([SceneRef](#) sceneRef, [NetworkLoadSceneParameters](#) sceneParams)
Gets the index of the given scene
- bool [RemoveSceneRef](#) ([SceneRef](#) sceneRef)
Removes a scene from the list
- override string [ToString](#) ()
String representation of the [NetworkSceneInfo](#)

Static Public Member Functions

- static implicit [operator NetworkSceneInfo](#) ([SceneRef](#) sceneRef)
Implicit conversion to [NetworkSceneInfo](#)

Static Public Attributes

- const int [MaxScenes](#) = 8
Max number of scenes that can be stored
- const int [SIZE](#) = 52
The size of the struct in bytes
- const int [WORD_COUNT](#) = 13
The size of the struct in words

Properties

- int [SceneCount](#) [get]
Total Scene Count
- [FixedArray](#)< [NetworkLoadSceneParameters](#) > [SceneParams](#) [get]
The scenes load parameters list
- [FixedArray](#)< [SceneRef](#) > [Scenes](#) [get]
The scenes list
- int [Version](#) [get]
Version number

6.259.1 Detailed Description

Can store up to 8 active scenes and allows for duplicates. Each write increases [Version](#) which can be used to generate unique scene objects ids for when a scene is supposed to be reloaded.

6.259.2 Member Function Documentation

6.259.2.1 AddSceneRef()

```
int AddSceneRef (
    SceneRef sceneRef,
    LoadSceneMode loadSceneMode = LoadSceneMode.Single,
    LocalPhysicsMode localPhysicsMode = LocalPhysicsMode.None,
    bool activeOnLoad = false )
```

Adds a scene to the list

Parameters

<i>sceneRef</i>	Scene to add
<i>loadSceneMode</i>	Load scene mode
<i>localPhysicsMode</i>	Local physics mode
<i>activeOnLoad</i>	Signals if the scene should be active on load

Returns

Returns the index of the scene or -1 if the scene could not be added

6.259.2.2 Equals() [1/2]

```
bool Equals (
    NetworkSceneInfo other )
```

Compares two [NetworkSceneInfo](#) for equality

Parameters

<i>other</i>	The other NetworkSceneInfo
--------------	--

Returns

Returns true if the two [NetworkSceneInfo](#) are equal

6.259.2.3 Equals() [2/2]

```
override bool Equals (
    object obj )
```

Compares two [NetworkSceneInfo](#) for equality

Parameters

<i>obj</i>	The other NetworkSceneInfo
------------	--

Returns

Returns true if the two [NetworkSceneInfo](#) are equal

6.259.2.4 GetHashCode()

```
override int GetHashCode ( )
```

Get the hash code of the [NetworkSceneInfo](#)

Returns

Hash code of the [NetworkSceneInfo](#)

6.259.2.5 IndexOf()

```
int IndexOf (
    SceneRef sceneRef,
    NetworkLoadSceneParameters sceneParams )
```

Gets the index of the given scene

Parameters

<i>sceneRef</i>	SceneRef to look for
<i>sceneParams</i>	Scene parameters to look for

Returns

Returns the index of the scene or -1 if not found

6.259.2.6 operator NetworkSceneInfo()

```
static implicit operator NetworkSceneInfo (
    SceneRef sceneRef ) [static]
```

Implicit conversion to [NetworkSceneInfo](#)

Parameters

<i>sceneRef</i>	SceneRef to convert
-----------------	---------------------

Returns

Returns a [NetworkSceneInfo](#) instance

6.259.2.7 RemoveSceneRef()

```
bool RemoveSceneRef (
    SceneRef sceneRef )
```

Removes a scene from the list

Parameters

<i>sceneRef</i>	Scene to remove
-----------------	-----------------

Returns

Returns true if the scene was removed

6.259.2.8 ToString()

```
override string ToString ( )
```

String representation of the [NetworkSceneInfo](#)

6.259.3 Member Data Documentation**6.259.3.1 MaxScenes**

```
const int MaxScenes = 8 [static]
```

Max number of scenes that can be stored

6.259.3.2 SIZE

```
const int SIZE = 52 [static]
```

The size of the struct in bytes

6.259.3.3 WORD_COUNT

```
const int WORD_COUNT = 13 [static]
```

The size of the struct in words

6.259.4 Property Documentation

6.259.4.1 SceneCount

```
int SceneCount [get]
```

Total Scene Count

6.259.4.2 SceneParams

```
FixedArray<NetworkLoadSceneParameters> SceneParams [get]
```

The scenes load parameters list

6.259.4.3 Scenes

```
FixedArray<SceneRef> Scenes [get]
```

The scenes list

6.259.4.4 Version

```
int Version [get]
```

Version number

6.260 NetworkSceneLoadId Struct Reference

A unique identifier for a scene load operation.

Inherits `IEquatable< NetworkSceneLoadId >`.

Public Member Functions

- bool `Equals` (`NetworkSceneLoadId` other)
Compares two `NetworkSceneLoadId` for equality
- override bool `Equals` (object obj)
Compares two `NetworkSceneLoadId` for equality
- override int `GetHashCode` ()
Returns the hash code of the `NetworkSceneLoadId`
- `NetworkSceneLoadId` (byte value)
Creates a new `NetworkSceneLoadId` with the given value
- override string `ToString` ()

Static Public Member Functions

- static implicit `operator NetworkSceneLoadId` (byte value)
- static bool `operator!=` (`NetworkSceneLoadId` left, `NetworkSceneLoadId` right)
- static bool `operator==` (`NetworkSceneLoadId` left, `NetworkSceneLoadId` right)

Public Attributes

- readonly byte `Value`
The value of the id

6.260.1 Detailed Description

A unique identifier for a scene load operation.

6.260.2 Constructor & Destructor Documentation

6.260.2.1 NetworkSceneLoadId()

```
NetworkSceneLoadId (  
    byte value )
```

Creates a new `NetworkSceneLoadId` with the given value

Parameters

<i>value</i>	The value of the id
--------------	---------------------

6.260.3 Member Function Documentation

6.260.3.1 Equals() [1/2]

```
bool Equals (
    NetworkSceneLoadId other )
```

Compares two [NetworkSceneLoadId](#) for equality

Parameters

<i>other</i>	The other NetworkSceneLoadId
--------------	--

Returns

Returns true if the two [NetworkSceneLoadId](#) are equal

6.260.3.2 Equals() [2/2]

```
override bool Equals (
    object obj )
```

Compares two [NetworkSceneLoadId](#) for equality

Parameters

<i>obj</i>	The other object to check
------------	---------------------------

Returns

Returns true if the two [NetworkSceneLoadId](#) are equal

6.260.3.3 GetHashCode()

```
override int GetHashCode ( )
```

Returns the hash code of the [NetworkSceneLoadId](#)

6.260.3.4 operator NetworkSceneLoadId()

```
static implicit operator NetworkSceneLoadId (  
    byte value ) [static]
```

Parameters

<i>value</i>	
--------------	--

Returns

6.260.3.5 operator"!=(())

```
static bool operator!= (  
    NetworkSceneLoadId left,  
    NetworkSceneLoadId right ) [static]
```

Parameters

<i>left</i>	
<i>right</i>	

Returns

6.260.3.6 operator==(())

```
static bool operator== (  
    NetworkSceneLoadId left,  
    NetworkSceneLoadId right ) [static]
```

Parameters

<i>left</i>	
<i>right</i>	

Returns

6.260.3.7 ToString()

```
override string ToString ( )
```

Returns

6.260.4 Member Data Documentation

6.260.4.1 Value

```
readonly byte Value
```

The value of the id

6.261 NetworkSceneObjectId Struct Reference

A unique identifier for a scene object.

Inherits `IEquatable< NetworkSceneObjectId >`.

Public Member Functions

- bool [Equals](#) ([NetworkSceneObjectId](#) other)
Check if two [NetworkSceneObjectId](#) are equal.
- override bool [Equals](#) (object obj)
Check if two [NetworkSceneObjectId](#) are equal.
- override int [GetHashCode](#) ()
Get the hash code of the [NetworkSceneObjectId](#).
- [NetworkSceneObjectId](#) ([SceneRef](#) scene, int objectId, [NetworkSceneLoadId](#) loadId=default)
TODO
- override string [ToString](#) ()
String representation of the [NetworkSceneObjectId](#).

Public Attributes

- [NetworkSceneLoadId](#) LoadId
Unique identifier of a specific scene load. Needs to be used when loading multiple scenes with the same [SceneRef](#) or reloading a scene. For example, [NetworkSceneInfo](#) increments its internal LoadId every time a new scene is added.
- int [ObjectId](#)
Index of the object in the scene or any other form of unique identifier.
- [SceneRef](#) Scene
Identifies the scene in which the object is located.

Properties

- bool `IsValid` [get]
Signal if the `NetworkSceneObjectId` is valid.
- int `SceneLoadId` [get]
Use `LoadId`

6.261.1 Detailed Description

A unique identifier for a scene object.

6.261.2 Constructor & Destructor Documentation

6.261.2.1 NetworkSceneObjectId()

```
NetworkSceneObjectId (
    SceneRef scene,
    int objectId,
    NetworkSceneLoadId loadId = default )
```

TODO

Parameters

<i>scene</i>	
<i>objectId</i>	
<i>loadId</i>	

6.261.3 Member Function Documentation

6.261.3.1 Equals() [1/2]

```
bool Equals (
    NetworkSceneObjectId other )
```

Check if two `NetworkSceneObjectId` are equal.

Parameters

<i>other</i>	Another <code>NetworkSceneObjectId</code> to check for equality
--------------	---

Returns

Returns true if the two [NetworkSceneObjectId](#) are equal

6.261.3.2 Equals() [2/2]

```
override bool Equals (  
    object obj )
```

Check if two [NetworkSceneObjectId](#) are equal.

Parameters

<i>obj</i>	Another NetworkSceneObjectId to check for equality
------------	--

Returns

Returns true if the two [NetworkSceneObjectId](#) are equal

6.261.3.3 GetHashCode()

```
override int GetHashCode ( )
```

Get the hash code of the [NetworkSceneObjectId](#).

6.261.3.4 ToString()

```
override string ToString ( )
```

String representation of the [NetworkSceneObjectId](#).

6.261.4 Member Data Documentation**6.261.4.1 LoadId**

[NetworkSceneLoadId](#) LoadId

Unique identifier of a specific scene load. Needs to be used when loading multiple scenes with the same [SceneRef](#) or reloading a scene. For example, [NetworkSceneInfo](#) increments its internal LoadId every time a new scene is added.

6.261.4.2 ObjectId

int ObjectId

Index of the object in the scene or any other form of unique identifier.

6.261.4.3 Scene

[SceneRef](#) Scene

Identifies the scene in which the object is located.

6.261.5 Property Documentation

6.261.5.1 IsValid

bool IsValid [get]

Signal if the [NetworkSceneObjectId](#) is valid.

6.261.5.2 SceneLoadId

int SceneLoadId [get]

Use [LoadId](#)

6.262 NetworkSerializeMethodAttribute Class Reference

Network Serialize Method Attribute

Inherits [Attribute](#).

Properties

- int [MaxSize](#) [get, set]

If set, this changes expected Wrap method signature to int Name(NetworkRunner, T, byte) and Unwrap to int Name(NetworkRunner, byte*, ref T). In both cases, the result is the number of bytes written/read and can not be greater than what's declared here.*

6.262.1 Detailed Description

Network Serialize Method Attribute

6.262.2 Property Documentation

6.262.2.1 MaxSize

```
int MaxSize [get], [set]
```

If set, this changes expected Wrap method signature to `int Name(NetworkRunner, T, byte*)` and Unwrap to `int Name(NetworkRunner, byte*, ref T)`. In both cases, the result is the number of bytes written/read and can not be greater than what's declared here.

6.263 NetworkSimulationConfiguration Class Reference

Configuration for network conditions simulation (induced latency and loss).

Inherits `IConfigurationSanityCheck`.

Public Member Functions

- [NetworkSimulationConfiguration Clone](#) ()
Creates a copy of this [NetworkSimulationConfiguration](#).
- [NetConfigSimulation Create](#) ()
Creates a new [NetConfigSimulation](#) based on the current configuration.
- void [SanityCheck](#) ()

Public Attributes

- double [AdditionalJitter](#) = 0.05
After the delay value from the [DelayShape](#) oscillator is determined, random 0 to this value of additional seconds be added to the packet latency.
- double [AdditionalLoss](#) = 0
After the [LossChanceShape](#) oscillation loss chance is calculated, an additional random value of 0 to this (normalized) percentage of loss chance is added.
- double [DelayMax](#) = 0.15
The highest packet delay value returned from the [DelayShape](#) oscillator.
- double [DelayMin](#) = 0.15
The lowest packet delay value returned from the [DelayShape](#) oscillator.
- double [DelayPeriod](#) = 0
The period of the [DelayShape](#) oscillator (the rate at which delay oscillates in seconds).
- [NetConfigSimulationOscillator.WaveShape DelayShape](#) = [NetConfigSimulationOscillator.WaveShape.Noise](#)
The pattern used to oscillate between [DelayMin](#) and [DelayMax](#) values.
- double [DelayThreshold](#) = 0

The *DelayShape* oscillates between 0 and 1. Values below this threshold are reduced to zero, resulting in a value equal to *DelayMin*.

- bool *Enabled*

If adverse network conditions are being simulated.

- double *LossChanceMax* = 0.05

The highest loss chance value the *LossChanceShape* oscillator will produce. 0 = 0% chance of being lost. 1 = 100% chance of being lost.

- double *LossChanceMin* = 0.05

The lowest loss chance value the *LossChanceShape* oscillator will produce. 0 = 0% chance of being lost. 1 = 100% chance of being lost.

- double *LossChancePeriod* = 0

The period of the *LossChanceShape* oscillator (the rate at which delay oscillates between *LossChanceMin* and *LossChanceMax*).

- *NetConfigSimulationOscillator.WaveShape* *LossChanceShape* = *NetConfigSimulationOscillator.WaveShape.Noise*

The pattern used to oscillate between *LossChanceMin* and *LossChanceMax* values.

- double *LossChanceThreshold* = 0

The *LossChanceShape* wave oscillates between 0 and 1. Values below this threshold are reduced to zero, resulting in a value equal to *LossChanceMin*.

6.263.1 Detailed Description

Configuration for network conditions simulation (induced latency and loss).

6.263.2 Member Function Documentation

6.263.2.1 Clone()

```
NetworkSimulationConfiguration Clone ( )
```

Creates a copy of this [NetworkSimulationConfiguration](#).

6.263.2.2 Create()

```
NetConfigSimulation Create ( )
```

Creates a new [NetConfigSimulation](#) based on the current configuration.

Returns

A new [NetConfigSimulation](#) based on the current configuration.

6.263.3 Member Data Documentation

6.263.3.1 AdditionalJitter

```
double AdditionalJitter = 0.05
```

After the delay value from the [DelayShape](#) oscillator is determined, random 0 to this value of additional seconds be added to the packet latency.

6.263.3.2 AdditionalLoss

```
double AdditionalLoss = 0
```

After the [LossChanceShape](#) oscillation loss chance is calculated, an additional random value of 0 to this (normalized) percentage of loss chance is added.

6.263.3.3 DelayMax

```
double DelayMax = 0.15
```

The highest packet delay value returned from the [DelayShape](#) oscillator.

6.263.3.4 DelayMin

```
double DelayMin = 0.15
```

The lowest packet delay value returned from the [DelayShape](#) oscillator.

6.263.3.5 DelayPeriod

```
double DelayPeriod = 0
```

The period of the [DelayShape](#) oscillator (the rate at which delay oscillates in seconds).

6.263.3.6 DelayShape

```
NetConfigSimulationOscillator.WaveShape DelayShape = NetConfigSimulationOscillator.Wave←  
Shape.Noise
```

The pattern used to oscillate between [DelayMin](#) and [DelayMax](#) values.

6.263.3.7 DelayThreshold

```
double DelayThreshold = 0
```

The [DelayShape](#) oscillates between 0 and 1. Values below this threshold are reduced to zero, resulting in a value equal to [DelayMin](#).

6.263.3.8 Enabled

```
bool Enabled
```

If adverse network conditions are being simulated.

6.263.3.9 LossChanceMax

```
double LossChanceMax = 0.05
```

The highest loss chance value the [LossChanceShape](#) oscillator will produce. 0 = 0% chance of being lost. 1 = 100% chance of being lost.

6.263.3.10 LossChanceMin

```
double LossChanceMin = 0.05
```

The lowest loss chance value the [LossChanceShape](#) oscillator will produce. 0 = 0% chance of being lost. 1 = 100% chance of being lost.

6.263.3.11 LossChancePeriod

```
double LossChancePeriod = 0
```

The period of the [LossChanceShape](#) oscillator (the rate at which delay oscillates between [LossChanceMin](#) and [LossChanceMax](#)).

6.263.3.12 LossChanceShape

```
NetConfigSimulationOscillator.WaveShape LossChanceShape = NetConfigSimulationOscillator.WaveShape.Noise
```

The pattern used to oscillate between [LossChanceMin](#) and [LossChanceMax](#) values.

6.263.3.13 LossChanceThreshold

```
double LossChanceThreshold = 0
```

The [LossChanceShape](#) wave oscillates between 0 and 1. Values below this threshold are reduced to zero, resulting in a value equal to [LossChanceMin](#).

6.264 NetworkSpawnOp Struct Reference

Spawn Operation

Classes

- struct [Awaiter](#)
Awaiter for [NetworkSpawnOp](#)

Public Member Functions

- [Awaiter GetAwaiter](#) ()
Get this Spawn Operation [Awaiter](#)

Public Attributes

- readonly [NetworkRunner Runner](#)
Network Runner Reference

Properties

- bool [IsFailed](#) [get]
Returns true if the object has failed to spawn.
- bool [IsQueued](#) [get]
Returns true if the object is still queued for spawning.
- bool [IsSpawned](#) [get]
Returns true if the object has been spawned.
- [NetworkObject Object](#) [get]
Get the spawned Network Object
- [NetworkSpawnStatus Status](#) [get]
Get the Spawn Operation Status

6.264.1 Detailed Description

Spawn Operation

6.264.2 Member Function Documentation

6.264.2.1 GetAwaiter()

```
Awaiter GetAwaiter ( )
```

Get this Spawn Operation [Awaiter](#)

6.264.3 Member Data Documentation

6.264.3.1 Runner

```
readonly NetworkRunner Runner
```

Network Runner Reference

6.264.4 Property Documentation

6.264.4.1 IsFailed

```
bool IsFailed [get]
```

Returns true if the object has failed to spawn.

6.264.4.2 IsQueued

```
bool IsQueued [get]
```

Returns true if the object is still queued for spawning.

6.264.4.3 IsSpawned

```
bool IsSpawned [get]
```

Returns true if the object has been spawned.

6.264.4.4 Object

[NetworkObject](#) Object [get]

Get the spawned Network Object

6.264.4.5 Status

[NetworkSpawnStatus](#) Status [get]

Get the Spawn Operation Status

6.265 NetworkSpawnOp.Awaiter Struct Reference

[Awaiter](#) for [NetworkSpawnOp](#)

Inherits [INotifyCompletion](#).

Public Member Functions

- [Awaiter](#) (in [NetworkSpawnOp](#) op)
Awaiter Constructor
- [NetworkObject GetResult](#) ()
Get the result of the Spawn Operation
- void [OnCompleted](#) (Action continuation)
Awaiter OnCompleted Callback

Public Attributes

- [NetworkSpawnOp _op](#)

Properties

- bool [IsCompleted](#) [get]
Returns true if the Spawn Operation is completed

6.265.1 Detailed Description

[Awaiter](#) for [NetworkSpawnOp](#)

6.265.2 Constructor & Destructor Documentation

6.265.2.1 Awaiter()

```
Awaiter (  
    in NetworkSpawnOp op )
```

[Awaiter](#) Constructor

Parameters

<i>op</i>	Spawn Operation
-----------	-----------------

6.265.3 Member Function Documentation

6.265.3.1 GetResult()

`NetworkObject` GetResult ()

Get the result of the Spawn Operation

Returns

Spawned Network Object

Exceptions

NetworkObjectSpawnException	Thrown if the Spawn Operation failed
---	--------------------------------------

6.265.3.2 OnCompleted()

```
void OnCompleted (
    Action continuation )
```

[Awaiter](#) OnCompleted Callback

Parameters

<i>continuation</i>	Continuation Action
---------------------	---------------------

Exceptions

NotSupportedException	Thrown if the Spawn Operation is not supported
---------------------------------------	--

6.265.4 Property Documentation

6.265.4.1 IsCompleted

```
bool IsCompleted [get]
```

Returns true if the Spawn Operation is completed

6.266 NetworkString< TSize > Class Template Reference

Fixed-size UTF32 string. All operations are alloc-free, except for converting to System.String.

Inherits INetworkString, [INetworkStruct](#), [IEquatable< NetworkString< TSize >>](#), and [IEnumerable< char >](#).

Public Member Functions

- void [Assign](#) (string value)
Assign a new value to this [NetworkString](#).
- int [Compare](#) ([NetworkString< TSize >](#) s)
Compares this instance with a specified [NetworkString](#).
- int [Compare](#) (ref [NetworkString< TSize >](#) s)
Compares this instance with a specified [NetworkString](#).
- int [Compare](#) (string s)
Compares this instance with a specified string.
- int [Compare< TOtherSize >](#) ([NetworkString< TOtherSize >](#) other)
Compares this instance with a specified [NetworkString](#) of a different size.
- int [Compare< TOtherSize >](#) (ref [NetworkString< TOtherSize >](#) other)
Compares this instance with a specified [NetworkString](#) of a different size.
- bool [Contains](#) (char c)
Determines whether a specified character is in this instance.
- bool [Contains](#) (string str)
Determines whether a specified string is in this instance.
- bool [Contains](#) (uint codePoint)
Determines whether a specified Unicode code point is in this instance.
- bool [Contains< TOtherSize >](#) ([NetworkString< TOtherSize >](#) str)
Determines whether a specified [NetworkString](#) is in this instance.
- bool [Contains< TOtherSize >](#) (ref [NetworkString< TOtherSize >](#) str)
Determines whether a specified [NetworkString](#) is in this instance.
- bool [EndsWith](#) (string s)
Checks if the current [NetworkString](#) ends with a specified string.
- bool [EndsWith< TOtherSize >](#) (ref [NetworkString< TOtherSize >](#) other)
Checks if the current [NetworkString](#) ends with a specified [NetworkString](#) of a different size.
- bool [Equals](#) ([NetworkString< TSize >](#) other)
Determines whether the current [NetworkString](#) is equal to a specified [NetworkString](#).
- override bool [Equals](#) (object obj)
Determines whether the current [NetworkString](#) is equal to a specified object.
- bool [Equals](#) (ref [NetworkString< TSize >](#) other)
Determines whether the current [NetworkString](#) is equal to a specified [NetworkString](#).
- bool [Equals](#) (string s)
Determines whether the current [NetworkString](#) is equal to a specified string.
- bool [Equals< TOtherSize >](#) ([NetworkString< TOtherSize >](#) other)

- Determines whether the current [NetworkString](#) is equal to a specified [NetworkString](#) of a different size.*

 - bool [Equals< TOtherSize >](#) (ref [NetworkString< TOtherSize >](#) other)
- Determines whether the current [NetworkString](#) is equal to a specified [NetworkString](#) of a different size.*

 - bool [Get](#) (ref string cache)
- Checks if cache is equivalent and if not converts to UTF16 and stores the result in cache .*

 - int [GetCharCount](#) ()
- Calculates the length of the equivalent UTF16 string.*

 - [UTF32Tools.CharEnumerator GetEnumerator](#) ()
- Returns an enumerator that iterates through the [NetworkString](#).*

 - IEnumerator< char > IEnumerable< char >. [GetEnumerator](#) ()
- Returns an enumerator that iterates through the [NetworkString](#).*

 - IEnumerable IEnumerable. [GetEnumerator](#) ()
- Returns the hash code for this [NetworkString](#).*

 - override int [GetHashCode](#) ()
- Returns the index of the first occurrence of a specified character in this instance.*

 - int [IndexOf](#) (char c, int startIndex, int count)
- Returns the index of the first occurrence of a specified character in this instance.*

 - int [IndexOf](#) (char c, int startIndex=0)
- Returns the index of the first occurrence of a specified string in this instance.*

 - int [IndexOf](#) (string str, int startIndex, int count)
- Returns the index of the first occurrence of a specified string in this instance.*

 - int [IndexOf](#) (string str, int startIndex=0)
- Returns the index of the first occurrence of a specified Unicode code point in this instance.*

 - int [IndexOf](#) (uint codePoint, int startIndex, int count)
- Returns the index of the first occurrence of a specified Unicode code point in this instance.*

 - int [IndexOf](#) (uint codePoint, int startIndex=0)
- Returns the index of the first occurrence of a specified [NetworkString](#) in this instance.*

 - int [IndexOf< TOtherSize >](#) ([NetworkString< TOtherSize >](#) str, int startIndex, int count)
- Returns the index of the first occurrence of a specified [NetworkString](#) in this instance.*

 - int [IndexOf< TOtherSize >](#) ([NetworkString< TOtherSize >](#) str, int startIndex=0)
- Returns the index of the first occurrence of a specified [NetworkString](#) in this instance.*

 - int [IndexOf< TOtherSize >](#) (ref [NetworkString< TOtherSize >](#) str, int startIndex, int count)
- Returns the index of the first occurrence of a specified [NetworkString](#) in this instance.*

 - int [IndexOf< TOtherSize >](#) (ref [NetworkString< TOtherSize >](#) str, int startIndex=0)
- Returns the index of the first occurrence of a specified [NetworkString](#) in this instance.*

 - [NetworkString](#) (string value)
- Creates a new instance of [NetworkString< Size >](#) with the given value.*

 - bool [Set](#) (string value)
- Converts value to UTF32 string and stores it internally.*

 - bool [StartsWith](#) (string s)
- Checks if the current [NetworkString](#) starts with a specified string.*

 - bool [StartsWith< TOtherSize >](#) (ref [NetworkString< TOtherSize >](#) other)
- Checks if the current [NetworkString](#) starts with a specified [NetworkString](#) of a different size.*

 - [NetworkString< TSize >](#) [Substring](#) (int startIndex)
- Returns a substring from this instance. The substring starts at a specified character position.*

 - [NetworkString< TSize >](#) [Substring](#) (int startIndex, int length)
- Returns a substring from this instance. The substring starts at a specified character position and has a specified length.*

 - [NetworkString< TSize >](#) [ToLower](#) ()
- Converts all the characters in this [NetworkString](#) to lowercase.*

 - override string [ToString](#) ()
- Converts the value of this [NetworkString](#) to its equivalent string representation.*

 - [NetworkString< TSize >](#) [ToUpper](#) ()
- Converts all the characters in this [NetworkString](#) to uppercase.*

Static Public Member Functions

- static int [GetCapacity< TSize > \(\)](#)
Gets the capacity of a [NetworkString](#) of a specified size.
- static implicit operator [NetworkString< TSize >](#) (string str)
Defines an implicit conversion of a string to a [NetworkString](#).
- static operator string ([NetworkString< TSize >](#) str)
Defines an explicit conversion of a [NetworkString](#) to a string.
- static bool operator!= ([NetworkString< TSize >](#) a, [NetworkString< TSize >](#) b)
Defines an inequality operator for [NetworkString](#).
- static bool operator!= ([NetworkString< TSize >](#) a, string b)
Defines an inequality operator for a [NetworkString](#) and a string.
- static bool operator!= (string a, [NetworkString< TSize >](#) b)
Defines an inequality operator for a string and a [NetworkString](#).
- static bool operator== ([NetworkString< TSize >](#) a, [NetworkString< TSize >](#) b)
Defines an equality operator for [NetworkString](#).
- static bool operator== ([NetworkString< TSize >](#) a, string b)
Defines an equality operator for a [NetworkString](#) and a string.
- static bool operator== (string a, [NetworkString< TSize >](#) b)
Defines an equality operator for a string and a [NetworkString](#).

Properties

- int [Capacity](#) [get]
Maximum UTF32 string length.
- int [Length](#) [get]
Number of UTF32 scalars. It is equal or less than [GetCharCount](#) or the length of [Value](#), because those use UTF16 encoding, which needs two characters to encode some values.
- ref uint [this\[int index\]](#) [get]
Returns UTF32 scalar at index position. To iterate over characters, use [GetEnumerator](#).
- string [Value](#) [get, set]
Converts to/from regular UTF16 string. Setter is alloc-free. Use [Get](#) to get possibly alloc-free conversion.

6.266.1 Detailed Description

Fixed-size UTF32 string. All operations are alloc-free, except for converting to System.String.

Provides static methods for [NetworkString](#) operations.

Template Parameters

TSize	
-----------------------	--

Type Constraints

***TSize* : [unmanaged](#)**

***TSize* : [IFixedStorage](#)**

6.266.2 Constructor & Destructor Documentation

6.266.2.1 NetworkString()

```
NetworkString (
    string value )
```

Creates a new instance of [NetworkString<Size>](#) with the given value.

Parameters

<i>value</i>	String value.
--------------	---------------

6.266.3 Member Function Documentation

6.266.3.1 Assign()

```
void Assign (
    string value )
```

Assign a new value to this [NetworkString](#).

Parameters

<i>value</i>	String value.
--------------	---------------

6.266.3.2 Compare() [1/3]

```
int Compare (
    NetworkString< TSize > s )
```

Compares this instance with a specified [NetworkString](#).

Parameters

<i>s</i>	The NetworkString to compare.
----------	---

Returns

A 32-bit signed integer that indicates the comparison result.

6.266.3.3 Compare() [2/3]

```
int Compare (
    ref NetworkString< TSize > s )
```

Compares this instance with a specified [NetworkString](#).

Parameters

s	The NetworkString to compare.
---	---

Returns

A 32-bit signed integer that indicates the comparison result.

6.266.3.4 Compare() [3/3]

```
int Compare (
    string s )
```

Compares this instance with a specified string.

Parameters

s	The string to compare.
---	------------------------

Returns

A 32-bit signed integer that indicates the comparison result.

6.266.3.5 Compare< TOtherSize >() [1/2]

```
int Compare< TOtherSize > (
    NetworkString< TOtherSize > other )
```

Compares this instance with a specified [NetworkString](#) of a different size.

Template Parameters

<i>TOtherSize</i>	The size of the other NetworkString .
-------------------	---

Parameters

<i>other</i>	The NetworkString to compare.
--------------	---

Returns

A 32-bit signed integer that indicates the comparison result.

Type Constraints

TOtherSize : unmanaged

TOtherSize : IFixedStorage

TOtherSize : Compare

TOtherSize : ref

TOtherSize : other

6.266.3.6 Compare< TOtherSize >() [2/2]

```
int Compare< TOtherSize > (  
    ref NetworkString< TOtherSize > other )
```

Compares this instance with a specified [NetworkString](#) of a different size.

Template Parameters

<i>TOtherSize</i>	The size of the other NetworkString .
-------------------	---

Parameters

<i>other</i>	The NetworkString to compare.
--------------	---

Returns

A 32-bit signed integer that indicates the comparison result.

Type Constraints

TOtherSize : unmanaged

TOtherSize : IFixedStorage

6.266.3.7 Contains() [1/3]

```
bool Contains (  
    char c )
```

Determines whether a specified character is in this instance.

Parameters

<i>c</i>	The Unicode character to seek.
----------	--------------------------------

Returns

true if the value parameter occurs within this string, or if value is the empty string (""); otherwise, false.

6.266.3.8 Contains() [2/3]

```
bool Contains (  
    string str )
```

Determines whether a specified string is in this instance.

Parameters

<i>str</i>	The string to seek.
------------	---------------------

Returns

true if the value parameter occurs within this string, or if value is the empty string (""); otherwise, false.

6.266.3.9 Contains() [3/3]

```
bool Contains (  
    uint codePoint )
```

Determines whether a specified Unicode code point is in this instance.

Parameters

<i>codePoint</i>	The Unicode code point to seek.
------------------	---------------------------------

Returns

true if the value parameter occurs within this string, or if value is the empty string (""); otherwise, false.

6.266.3.10 Contains< TOtherSize >() [1/2]

```
bool Contains< TOtherSize > (  
    NetworkString< TOtherSize > str )
```

Determines whether a specified [NetworkString](#) is in this instance.

Template Parameters

<i>TOtherSize</i>	The size of the other NetworkString .
-------------------	---

Parameters

<i>str</i>	The NetworkString to seek.
------------	--

Returns

true if the value parameter occurs within this string, or if value is the empty string (""); otherwise, false.

Type Constraints

TOtherSize : unmanaged

TOtherSize : IFixedStorage

TOtherSize : IndexOf

TOtherSize : ref

TOtherSize : str

6.266.3.11 Contains< TOtherSize >() [2/2]

```
bool Contains< TOtherSize > (  
    ref NetworkString< TOtherSize > str )
```

Determines whether a specified [NetworkString](#) is in this instance.

Template Parameters

<i>TOtherSize</i>	The size of the other NetworkString .
-------------------	---

Parameters

<i>str</i>	The NetworkString to seek.
------------	--

Returns

true if the value parameter occurs within this string, or if value is the empty string (""); otherwise, false.

Type Constraints

TOtherSize : *unmanaged*

TOtherSize : *IFixedStorage*

TOtherSize : *IndexOf*

TOtherSize : *ref*

TOtherSize : *str*

6.266.3.12 EndsWith()

```
bool EndsWith (
    string s )
```

Checks if the current [NetworkString](#) ends with a specified string.

Parameters

s	The string to check.
---	----------------------

Returns

true if the current [NetworkString](#) ends with the specified string; otherwise, false.

Exceptions

<i>ArgumentNullException</i>	Thrown when the string is null.
------------------------------	---------------------------------

6.266.3.13 EndsWith< TOtherSize >()

```
bool EndsWith< TOtherSize > (
    ref NetworkString< TOtherSize > other )
```

Checks if the current [NetworkString](#) ends with a specified [NetworkString](#) of a different size.

Template Parameters

<i>TOtherSize</i>	The size of the other NetworkString .
-------------------	---

Parameters

<i>other</i>	The NetworkString to check.
--------------	---

Returns

true if the current [NetworkString](#) ends with the specified [NetworkString](#); otherwise, false.

Type Constraints

TOtherSize* : *unmanaged

TOtherSize* : *IFixedStorage

6.266.3.14 Equals() [1/4]

```
bool Equals (
    NetworkString< TSize > other )
```

Determines whether the current [NetworkString](#) is equal to a specified [NetworkString](#).

Parameters

<i>other</i>	The NetworkString to compare with the current NetworkString .
--------------	---

Returns

true if the specified [NetworkString](#) is equal to the current [NetworkString](#); otherwise, false.

6.266.3.15 Equals() [2/4]

```
override bool Equals (
    object obj )
```

Determines whether the current [NetworkString](#) is equal to a specified object.

Parameters

<i>obj</i>	The object to compare with the current NetworkString .
------------	--

Returns

true if the specified object is equal to the current [NetworkString](#); otherwise, false.

6.266.3.16 Equals() [3/4]

```
bool Equals (
    ref NetworkString< TSize > other )
```

Determines whether the current [NetworkString](#) is equal to a specified [NetworkString](#).

Parameters

<i>other</i>	The NetworkString to compare with the current NetworkString .
--------------	---

Returns

true if the specified [NetworkString](#) is equal to the current [NetworkString](#); otherwise, false.

6.266.3.17 Equals() [4/4]

```
bool Equals (
    string s )
```

Determines whether the current [NetworkString](#) is equal to a specified string.

Parameters

<i>s</i>	The string to compare with the current NetworkString .
----------	--

Returns

true if the specified string is equal to the current [NetworkString](#); otherwise, false.

6.266.3.18 Equals< TOtherSize >() [1/2]

```
bool Equals< TOtherSize > (
    NetworkString< TOtherSize > other )
```

Determines whether the current [NetworkString](#) is equal to a specified [NetworkString](#) of a different size.

Template Parameters

<i>TOtherSize</i>	The size of the other NetworkString .
-------------------	---

Parameters

<i>other</i>	The NetworkString to compare with the current NetworkString .
--------------	---

Returns

true if the specified [NetworkString](#) is equal to the current [NetworkString](#); otherwise, false.

Type Constraints

TOtherSize : unmanaged

TOtherSize : IFixedStorage

TOtherSize : Compare

TOtherSize : ref

TOtherSize : other

6.266.3.19 Equals< TOtherSize >() [2/2]

```
bool Equals< TOtherSize > (
    ref NetworkString< TOtherSize > other )
```

Determines whether the current [NetworkString](#) is equal to a specified [NetworkString](#) of a different size.

Template Parameters

<i>TOtherSize</i>	The size of the other NetworkString .
-------------------	---

Parameters

<i>other</i>	The NetworkString to compare with the current NetworkString .
--------------	---

Returns

true if the specified [NetworkString](#) is equal to the current [NetworkString](#); otherwise, false.

Type Constraints

TOtherSize : unmanaged

TOtherSize : IFixedStorage

TOtherSize : Compare

TOtherSize : ref

TOtherSize : other

6.266.3.20 Get()

```
bool Get (
    ref string cache )
```

Checks if *cache* is equivalent and if not converts to UTF16 and stores the result in *cache* .

Parameters

<code>cache</code>	The string to convert.
--------------------	------------------------

Returns

False if no conversion was performed, true otherwise.

6.266.3.21 GetCapacity< TSize >()

```
static int GetCapacity< TSize > ( ) [static]
```

Gets the capacity of a [NetworkString](#) of a specified size.

Template Parameters

<code>TSize</code>	The size of the NetworkString .
--------------------	---

Returns

The capacity of a [NetworkString](#) of the specified size.

Type Constraints

TSize : unmanaged

TSize : IFixedStorage

6.266.3.22 GetCharCount()

```
int GetCharCount ( )
```

Calculates the length of the equivalent UTF16 string.

Returns

The length of the equivalent UTF16 string.

6.266.3.23 GetEnumerator()

```
UTF32Tools.CharEnumerator GetEnumerator ( )
```

Returns an enumerator that iterates through the [NetworkString](#).

Returns

A [UTF32Tools.CharEnumerator](#) for the [NetworkString](#).

6.266.3.24 GetHashCode()

```
override int GetHashCode ( )
```

Returns the hash code for this [NetworkString](#).

Returns

A 32-bit signed integer hash code.

6.266.3.25 IndexOf() [1/6]

```
int IndexOf (
    char c,
    int startIndex,
    int count )
```

Returns the index of the first occurrence of a specified character in this instance.

Parameters

<i>c</i>	The Unicode character to seek.
<i>startIndex</i>	The search starting position.
<i>count</i>	The number of character positions to examine.

Returns

The zero-based index position of value if that character is found, or -1 if it is not.

6.266.3.26 IndexOf() [2/6]

```
int IndexOf (
    char c,
    int startIndex = 0 )
```

Returns the index of the first occurrence of a specified character in this instance.

Parameters

<i>c</i>	The Unicode character to seek.
<i>startIndex</i>	The search starting position.

Returns

The zero-based index position of value if that character is found, or -1 if it is not.

6.266.3.27 IndexOf() [3/6]

```
int IndexOf (
    string str,
    int startIndex,
    int count )
```

Returns the index of the first occurrence of a specified string in this instance.

Parameters

<i>str</i>	The string to seek.
<i>startIndex</i>	The search starting position.
<i>count</i>	The number of character positions to examine.

Returns

The zero-based index position of value if that string is found, or -1 if it is not.

Exceptions

<i>ArgumentNullException</i>	Thrown when the string is null.
<i>ArgumentOutOfRangeException</i>	Thrown when the start index is less than zero or greater than the safe length of the string, or when the count is less than zero or the sum of the start index and count is greater than the safe length of the string.

6.266.3.28 IndexOf() [4/6]

```
int IndexOf (
    string str,
    int startIndex = 0 )
```

Returns the index of the first occurrence of a specified string in this instance.

Parameters

<i>str</i>	The string to seek.
<i>startIndex</i>	The search starting position.

Returns

The zero-based index position of value if that string is found, or -1 if it is not.

6.266.3.29 IndexOf() [5/6]

```
int IndexOf (
    uint codePoint,
    int startIndex,
    int count )
```

Returns the index of the first occurrence of a specified Unicode code point in this instance.

Parameters

<i>codePoint</i>	The Unicode code point to seek.
<i>startIndex</i>	The search starting position.
<i>count</i>	The number of character positions to examine.

Returns

The zero-based index position of value if that Unicode code point is found, or -1 if it is not.

Exceptions

<i>ArgumentOutOfRangeException</i>	Thrown when the start index is less than zero or greater than the safe length of the string, or when the count is less than zero or the sum of the start index and count is greater than the safe length of the string.
------------------------------------	---

6.266.3.30 IndexOf() [6/6]

```
int IndexOf (
    uint codePoint,
    int startIndex = 0 )
```

Returns the index of the first occurrence of a specified Unicode code point in this instance.

Parameters

<i>codePoint</i>	The Unicode code point to seek.
<i>startIndex</i>	The search starting position.

Returns

The zero-based index position of value if that Unicode code point is found, or -1 if it is not.

6.266.3.31 IndexOf< TOtherSize >() [1/4]

```
int IndexOf< TOtherSize > (
    NetworkString< TOtherSize > str,
    int startIndex,
    int count )
```

Returns the index of the first occurrence of a specified [NetworkString](#) in this instance.

Template Parameters

<i>TOtherSize</i>	The size of the other NetworkString .
-------------------	---

Parameters

<i>str</i>	The NetworkString to seek.
<i>startIndex</i>	The search starting position.
<i>count</i>	The number of character positions to examine.

Returns

The zero-based index position of value if that [NetworkString](#) is found, or -1 if it is not.

Type Constraints

TOtherSize : unmanaged
TOtherSize : IFixedStorage
TOtherSize : IndexOf
TOtherSize : ref
TOtherSize : str
TOtherSize : startIndex
TOtherSize : count

6.266.3.32 IndexOf< TOtherSize >() [2/4]

```
int IndexOf< TOtherSize > (
    NetworkString< TOtherSize > str,
    int startIndex = 0 )
```

Returns the index of the first occurrence of a specified [NetworkString](#) in this instance.

Template Parameters

<i>TOtherSize</i>	The size of the other NetworkString .
-------------------	---

Parameters

<i>str</i>	The NetworkString to seek.
<i>startIndex</i>	The search starting position.

Returns

The zero-based index position of value if that [NetworkString](#) is found, or -1 if it is not.

Type Constraints

TOtherSize : unmanaged

TOtherSize : IFixedStorage

TOtherSize : IndexOf

TOtherSize : ref

TOtherSize : str

TOtherSize : startIndex

TOtherSize : SafeLength

TOtherSize : startIndex

6.266.3.33 IndexOf< TOtherSize >() [3/4]

```
int IndexOf< TOtherSize > (
    ref NetworkString< TOtherSize > str,
    int startIndex,
    int count )
```

Returns the index of the first occurrence of a specified [NetworkString](#) in this instance.

Template Parameters

<i>TOtherSize</i>	The size of the other NetworkString .
-------------------	---

Parameters

<i>str</i>	The NetworkString to seek.
<i>startIndex</i>	The search starting position.
<i>count</i>	The number of character positions to examine.

Returns

The zero-based index position of value if that [NetworkString](#) is found, or -1 if it is not.

Exceptions

<i>ArgumentOutOfRangeException</i>	Thrown when the start index is less than zero or greater than the safe length of the string, or when the count is less than zero or the sum of the start index and count is greater than the safe length of the string.
------------------------------------	---

Type Constraints

TOtherSize : unmanaged

TOtherSize : IFixedStorage

6.266.3.34 IndexOf< TOtherSize >() [4/4]

```
int IndexOf< TOtherSize > (
    ref NetworkString< TOtherSize > str,
    int startIndex = 0 )
```

Returns the index of the first occurrence of a specified [NetworkString](#) in this instance.

Template Parameters

<i>TOtherSize</i>	The size of the other NetworkString .
-------------------	---

Parameters

<i>str</i>	The NetworkString to seek.
<i>startIndex</i>	The search starting position.

Returns

The zero-based index position of value if that [NetworkString](#) is found, or -1 if it is not.

Type Constraints

TOtherSize : unmanaged

TOtherSize : IFixedStorage

TOtherSize : IndexOf

TOtherSize : ref

TOtherSize : str

TOtherSize : startIndex

TOtherSize : SafeLength

TOtherSize : startIndex

6.266.3.35 operator NetworkString< TSize >()

```
static implicit operator NetworkString< TSize > (  
    string str ) [static]
```

Defines an implicit conversion of a string to a [NetworkString](#).

Parameters

<i>str</i>	The string to convert.
------------	------------------------

Returns

A new instance of [NetworkString](#) with the same value as the string.

6.266.3.36 operator string()

```
static operator string (  
    NetworkString< TSize > str ) [explicit], [static]
```

Defines an explicit conversion of a [NetworkString](#) to a string.

Parameters

<i>str</i>	The NetworkString to convert.
------------	---

Returns

The string value of the [NetworkString](#).

6.266.3.37 operator"!=(()) [1/3]

```
static bool operator!= (  
    NetworkString< TSize > a,  
    NetworkString< TSize > b ) [static]
```

Defines an inequality operator for [NetworkString](#).

Parameters

<i>a</i>	The first NetworkString to compare.
<i>b</i>	The second NetworkString to compare.

Returns

true if the NetworkStrings are not equal; otherwise, false.

6.266.3.38 operator"!="() [2/3]

```
static bool operator!= (
    NetworkString< TSize > a,
    string b ) [static]
```

Defines an inequality operator for a [NetworkString](#) and a string.

Parameters

<i>a</i>	The NetworkString to compare.
<i>b</i>	The string to compare.

Returns

true if the [NetworkString](#) and the string are not equal; otherwise, false.

6.266.3.39 operator"!="() [3/3]

```
static bool operator!= (
    string a,
    NetworkString< TSize > b ) [static]
```

Defines an inequality operator for a string and a [NetworkString](#).

Parameters

<i>a</i>	The string to compare.
<i>b</i>	The NetworkString to compare.

Returns

true if the string and the [NetworkString](#) are not equal; otherwise, false.

6.266.3.40 operator==() [1/3]

```
static bool operator==(
    NetworkString< TSize > a,
    NetworkString< TSize > b ) [static]
```

Defines an equality operator for [NetworkString](#).

Parameters

<i>a</i>	The first NetworkString to compare.
<i>b</i>	The second NetworkString to compare.

Returns

true if the [NetworkStrings](#) are equal; otherwise, false.

6.266.3.41 operator==() [2/3]

```
static bool operator==(
    NetworkString< TSize > a,
    string b ) [static]
```

Defines an equality operator for a [NetworkString](#) and a string.

Parameters

<i>a</i>	The NetworkString to compare.
<i>b</i>	The string to compare.

Returns

true if the [NetworkString](#) and the string are equal; otherwise, false.

6.266.3.42 operator==() [3/3]

```
static bool operator==(
    string a,
    NetworkString< TSize > b ) [static]
```

Defines an equality operator for a string and a [NetworkString](#).

Parameters

<i>a</i>	The string to compare.
<i>b</i>	The NetworkString to compare.

Returns

true if the string and the [NetworkString](#) are equal; otherwise, false.

6.266.3.43 Set()

```
bool Set (
    string value )
```

Converts *value* to UTF32 string and stores it internally.

Parameters

<i>value</i>	The string to set.
--------------	--------------------

Returns

False if *value* was too long to fit and had to be trimmed.

6.266.3.44 StartsWith()

```
bool StartsWith (
    string s )
```

Checks if the current [NetworkString](#) starts with a specified string.

Parameters

<i>s</i>	The string to check.
----------	----------------------

Returns

true if the current [NetworkString](#) starts with the specified string; otherwise, false.

Exceptions

<i>ArgumentNullException</i>	Thrown when the string is null.
------------------------------	---------------------------------

6.266.3.45 StartsWith< TOtherSize >()

```
bool StartsWith< TOtherSize > (
    ref NetworkString< TOtherSize > other )
```

Checks if the current [NetworkString](#) starts with a specified [NetworkString](#) of a different size.

Template Parameters

<i>TOtherSize</i>	The size of the other NetworkString .
-------------------	---

Parameters

<i>other</i>	The NetworkString to check.
--------------	---

Returns

true if the current [NetworkString](#) starts with the specified [NetworkString](#); otherwise, false.

Type Constraints

TOtherSize* : *unmanaged

TOtherSize* : *IFixedStorage

6.266.3.46 Substring() [1/2]

```
NetworkString<TSize> Substring (
    int startIndex )
```

Returns a substring from this instance. The substring starts at a specified character position.

Parameters

<i>startIndex</i>	The zero-based starting character position of a substring in this instance.
-------------------	---

Returns

A new [NetworkString](#) that is equivalent to the substring that begins at *startIndex* in this instance, or [NetworkString.Empty](#) if *startIndex* is equal to the length of this instance.

Exceptions

<i>ArgumentOutOfRangeException</i>	<i>startIndex</i> is less than zero or greater than the length of this instance.
------------------------------------	--

6.266.3.47 Substring() [2/2]

```
NetworkString<TSize> Substring (
    int startIndex,
    int length )
```

Returns a substring from this instance. The substring starts at a specified character position and has a specified length.

Parameters

<i>startIndex</i>	The zero-based starting character position of a substring in this instance.
<i>length</i>	The number of characters in the substring.

Returns

A new [NetworkString](#) that is equivalent to the substring of length `length` that begins at `startIndex` in this instance, or `NetworkString.Empty` if `startIndex` is equal to the length of this instance and `length` is zero.

Exceptions

<i>ArgumentOutOfRangeException</i>	<code>startIndex</code> plus <code>length</code> indicates a position not within this instance, or <code>startIndex</code> or <code>length</code> is less than zero.
------------------------------------	--

6.266.3.48 ToLower()

```
NetworkString<TSize> ToLower ( )
```

Converts all the characters in this [NetworkString](#) to lowercase.

Returns

A new [NetworkString](#) in which all characters in this [NetworkString](#) are converted to lowercase.

6.266.3.49 ToString()

```
override string ToString ( )
```

Converts the value of this [NetworkString](#) to its equivalent string representation.

Returns

A string representation of the value of this [NetworkString](#).

6.266.3.50 ToUpper()

```
NetworkString<TSize> ToUpper ( )
```

Converts all the characters in this [NetworkString](#) to uppercase.

Returns

A new [NetworkString](#) in which all characters in this [NetworkString](#) are converted to uppercase.

6.266.4 Property Documentation

6.266.4.1 Capacity

```
int Capacity [get]
```

Maximum UTF32 string length.

6.266.4.2 Length

```
int Length [get]
```

Number of UTF32 scalars. It is equal or less than [GetCharCount](#) or the length of [Value](#), because those use UTF16 encoding, which needs two characters to encode some values.

6.266.4.3 this[int index]

```
ref uint this[int index] [get]
```

Returns UTF32 scalar at *index* position. To iterate over characters, use [GetEnumerator](#).

Parameters

<i>index</i>	Index to get.
--------------	---------------

Returns

UTF32 scalar at *index* position.

6.266.4.4 Value

```
string Value [get], [set]
```

Converts to/from regular UTF16 string. Setter is alloc-free. Use [Get](#) to get possibly alloc-free conversion.

6.267 NetworkStructUtils Class Reference

Utility methods for [INetworkStruct](#)

Static Public Member Functions

- static int [GetWordCount](#) (Type type)
- static int [GetWordCount< T > \(\)](#)
Get *INetworkStruct* Word Count

6.267.1 Detailed Description

Utility methods for [INetworkStruct](#)

6.267.2 Member Function Documentation

6.267.2.1 [GetWordCount< T >\(\)](#)

```
static int GetWordCount< T > \(\) [static]
```

Get [INetworkStruct](#) Word Count

Template Parameters

<i>T</i>	INetworkStruct type reference
----------	---

Returns

Number of Words necessary for this specific [INetworkStruct](#)

Type Constraints

T: *unmanaged*

T: *INetworkStruct*

6.268 NetworkStructWeavedAttribute Class Reference

Describes the total number of WORDs a [INetworkStruct](#) uses.

Inherits Attribute.

Public Member Functions

- [NetworkStructWeavedAttribute](#) (int wordCount)
NetworkStructWeavedAttribute Constructor

Properties

- int [WordCount](#) [get]
INetworkStruct Word Count

6.268.1 Detailed Description

Describes the total number of WORDs a [INetworkStruct](#) uses.

6.268.2 Constructor & Destructor Documentation

6.268.2.1 NetworkStructWeavedAttribute()

```
NetworkStructWeavedAttribute (
    int wordCount )
```

[NetworkStructWeavedAttribute](#) Constructor

Parameters

<i>wordCount</i>	INetworkStruct word count
------------------	---

6.268.3 Property Documentation

6.268.3.1 WordCount

```
int WordCount [get]
```

[INetworkStruct](#) Word Count

6.269 NetworkTransform Class Reference

Add to any [NetworkObject](#) Transform, or its associated child Transforms to automatically synchronize TRSP (Position/Rotation/Scale/Parent).

Inherits [NetworkTRSP](#), [INetworkTRSPTeleport](#), [IBeforeAllTicks](#), [IAfterAllTicks](#), and [IBeforeCopyPreviousState](#).

Public Member Functions

- override void [Render](#) ()
Post simulation frame rendering callback. Runs after all simulations have finished. Use in place of Unity's Update when [Fusion](#) is handling Physics.
- override void [SetAreaOfInterestOverride](#) ([NetworkObject](#) obj)
Manually set the [NetworkObject](#) used as the AreaOfInterestOverride.
- override void [Spawned](#) ()
Post spawn callback.
- void [Teleport](#) (Vector3? position=null, Quaternion? rotation=null)
Set the transform position and rotation to the indicated values, and network the Teleport event. This will suspend interpolation between the previous tick state and the current tick state in [Render\(\)](#), on this peer and all remote peers.

Public Attributes

- bool [DisableSharedModelInterpolation](#) = false
Disable interpolation on State Authority in Shared Mode. You should disable interpolation if your controller code moves an object inside of Update() rather than [FixedUpdateNetwork\(\)](#).
- bool [SyncParent](#) = false
Enables synchronization of transform.parent. NOTE: Parent GameObjects must have a [NetworkBehaviour](#) derived component to be a valid parent, parent must belong to a different [NetworkObject](#) than this Object.
- bool [SyncScale](#) = false
Enables synchronization of LocalScale.

Properties

- bool [AutoUpdateAreaOfInterestOverride](#) [get, set]
Determines if parent changes should automatically call [SetAreaOfInterestOverride\(NetworkObject\)](#), and assign the parent [NetworkObject](#) as the override. Default is true, as you typically will want player interest in this object to reflect player interest in the nested parent object. For example, if a player is carrying an nested Object, players should only see that carried Object if they see the player. Additionally, AOI works in world space, and [NetworkTransform](#) operates in local space, so any AOI position values of nested Objects will ALWAYS be invalid, so nested Objects should always have their AOI Override set to a non-nested Object.

Additional Inherited Members

6.269.1 Detailed Description

Add to any [NetworkObject](#) Transform, or its associated child Transforms to automatically synchronize TRSP (Position/Rotation/Scale/Parent).

6.269.2 Member Function Documentation

6.269.2.1 SetAreaOfInterestOverride()

```
override void SetAreaOfInterestOverride (
    NetworkObject obj ) [virtual]
```

Manually set the [NetworkObject](#) used as the AreaOfInterestOverride.

Parameters

<i>obj</i>	NetworkObject to use as the AreaOfInterestOverride.
------------	---

Reimplemented from [NetworkTRSP](#).

6.269.2.2 Teleport()

```
void Teleport (
    Vector3? position = null,
    Quaternion? rotation = null )
```

Set the transform position and rotation to the indicated values, and network the Teleport event. This will suspend interpolation between the previous tick state and the current tick state in [Render\(\)](#), on this peer and all remote peers.

Implements [INetworkTRSPTeleport](#).

6.269.3 Member Data Documentation

6.269.3.1 DisableSharedModeInterpolation

```
bool DisableSharedModeInterpolation = false
```

Disable interpolation on State Authority in Shared Mode. You should disable interpolation if your controller code moves an object inside of Update() rather than [FixedUpdateNetwork\(\)](#).

6.269.3.2 SyncParent

```
bool SyncParent = false
```

Enables synchronization of transform.parent. NOTE: Parent GameObjects must have a [NetworkBehaviour](#) derived component to be a valid parent, parent must belong to a different [NetworkObject](#) than this Object.

6.269.3.3 SyncScale

```
bool SyncScale = false
```

Enables synchronization of LocalScale.

6.269.4 Property Documentation

6.269.4.1 AutoUpdateAreaOfInterestOverride

```
bool AutoUpdateAreaOfInterestOverride [get], [set]
```

Determines if parent changes should automatically call [SetAreaOfInterestOverride\(NetworkObject\)](#), and assign the parent [NetworkObject](#) as the override. Default is true, as you typically will want player interest in this object to reflect player interest in the nested parent object. For example, if a player is carrying an nested Object, players should only see that carried Object if they see the player. Additionally, AOI works in world space, and [NetworkTransform](#) operates in local space, so any AOI position values of nested Objects will ALWAYS be invalid, so nested Objects should always have their AOI Override set to a non-nested Object.

6.270 NetworkTRSP Class Reference

Base class for spatial (Position/Rotation/Scale/Parent) synchronization component, such as [NetworkTransform](#). Provides the base logic for render interpolation, parenting synchronization, and teleport, that can be used in components derived from this class.

Inherits [NetworkBehaviour](#).

Inherited by [NetworkTransform](#).

Public Member Functions

- virtual void [SetAreaOfInterestOverride](#) ([NetworkObject](#) obj)
Manually set the [NetworkObject](#) used as the AreaOfInterestOverride.

Static Protected Member Functions

- static void [Render](#) ([NetworkTRSP](#) behaviour, Transform transform, bool syncScale, bool syncParent, bool local, ref [Tick](#) initial)
Default Render handling for [NetworkTRSP](#) derived classes.
- static void [ResolveAOIOVERRIDE](#) ([NetworkTRSP](#) behaviour, Transform parent)
Recursively attempts to find nested parent [NetworkObject](#), and if found assigns that [NetworkObject](#) as the AreaOfInterestOverride.
- static void [SetParentTransform](#) ([NetworkTRSP](#) behaviour, Transform transform, [NetworkBehaviourId](#) parent↔Id)
Default handling for setting a [NetworkTRSP](#)'s parent using a [NetworkBehaviourId](#) value.
- static void [Teleport](#) ([NetworkTRSP](#) behaviour, Transform transform, Vector3? position=null, Quaternion? rotation=null)
The default Teleport implementation for [NetworkTRSP](#) derived classes.

Protected Attributes

- [Tick](#) [reenabledTick](#)
The tick on which this [NetworkTRSP](#) was last re-enabled (right after Spawned doesn't count).

Properties

- [NetworkTRSPData Data](#) [get]
The networked data of this [NetworkTRSP](#).
- bool [IsMainTRSP](#) [get]
The main [NetworkTRSP](#) is at the root of the [NetworkObject](#) and it will be used for area of interest operations and parenting of the [NetworkObject](#).
- ref [NetworkTRSPData State](#) [get]
A reference to the networked data of this [NetworkTRSP](#).

Additional Inherited Members

6.270.1 Detailed Description

Base class for spatial (Position/Rotation/Scale/Parent) synchronization component, such as [NetworkTransform](#). Provides the base logic for render interpolation, parenting synchronization, and teleport, that can be used in components derived from this class.

6.270.2 Member Function Documentation

6.270.2.1 Render()

```
static void Render (
    NetworkTRSP behaviour,
    Transform transform,
    bool syncScale,
    bool syncParent,
    bool local,
    ref Tick initial ) [static], [protected]
```

Default Render handling for [NetworkTRSP](#) derived classes.

6.270.2.2 ResolveAOIOVERRIDE()

```
static void ResolveAOIOVERRIDE (
    NetworkTRSP behaviour,
    Transform parent ) [static], [protected]
```

Recursively attempts to find nested parent [NetworkObject](#), and if found assigns that [NetworkObject](#) as the [AreaOfInterestOverride](#).

Parameters

<i>behaviour</i>	Only pass a NetworkTRSP derived class that is on the same Transform as its associated NetworkObject , as AreaOfInterestOverride is only applicable when IsMainTRSP is true.
------------------	---

Parameters

<i>parent</i>	The direct parent of the
---------------	--------------------------

6.270.2.3 SetAreaOfInterestOverride()

```
virtual void SetAreaOfInterestOverride (
    NetworkObject obj ) [virtual]
```

Manually set the [NetworkObject](#) used as the AreaOfInterestOverride.

Parameters

<i>obj</i>	NetworkObject to use as the AreaOfInterestOverride.
------------	---

Reimplemented in [NetworkTransform](#).

6.270.2.4 SetParentTransform()

```
static void SetParentTransform (
    NetworkTRSP behaviour,
    Transform transform,
    NetworkBehaviourId parentId ) [static], [protected]
```

Default handling for setting a [NetworkTRSP](#)'s parent using a [NetworkBehaviourId](#) value.

6.270.2.5 Teleport()

```
static void Teleport (
    NetworkTRSP behaviour,
    Transform transform,
    Vector3? position = null,
    Quaternion? rotation = null ) [static], [protected]
```

The default Teleport implementation for [NetworkTRSP](#) derived classes.

6.270.3 Member Data Documentation

6.270.3.1 reenabledTick

`Tick` reenabledTick [protected]

The tick on which this [NetworkTRSP](#) was last re-enabled (right after Spawned doesn't count).

Used to override interpolation alpha for the tick after being re-enabled.

6.270.4 Property Documentation

6.270.4.1 Data

`NetworkTRSPData` Data [get]

The networked data of this [NetworkTRSP](#).

6.270.4.2 IsMainTRSP

`bool` IsMainTRSP [get]

The main [NetworkTRSP](#) is at the root of the [NetworkObject](#) and it will be used for area of interest operations and parenting of the [NetworkObject](#).

6.270.4.3 State

`ref` `NetworkTRSPData` State [get], [protected]

A reference to the networked data of this [NetworkTRSP](#).

6.271 NetworkTRSPData Struct Reference

Data structure storing spatial (Position/Rotation/Scale/Parent) synchronization data for spatial synchronization components, [NetworkTRSP](#) and its subclass [NetworkTransform](#).

Inherits [INetworkStruct](#).

Public Attributes

- [NetworkId AreaOfInterestOverride](#)
Id of a behaviour used as the reference point for this component during area of interest operations The behaviour should be a [NetworkTRSP](#) derived class, that is on the same Transform as its associated [NetworkObject](#)
- [NetworkBehaviourId Parent](#)
Id of a [NetworkBehaviour](#) on the parent of the component's transform.
- [Vector3 Position](#)
Position relevant for the spatial synchronization component (can be used to either store a local position or a world position, depending on the component)
- [Quaternion Rotation](#)
Rotation relevant for the spatial synchronization component (can be used to either store a local rotation or a world rotation, depending on the component)
- [Vector3Compressed Scale](#)
Scale relevant for the spatial synchronization component
- [int TeleportKey](#)
Key used to differentiate between several teleports

Static Public Attributes

- `const int POSITION_OFFSET = 2`
Offset to point at the position values on the data buffer
- `const int SIZE = WORDS * Allocator.REPLICATE_WORD_SIZE`
The actual size for the networked properties in bytes
- `const int WORDS = 14`
Networked properties word count for the base [NetworkTRSPData](#)

Properties

- `static NetworkBehaviourId NonNetworkedParent [get]`
Special [NetworkBehaviourId](#) value, used as a flag to tell the parent is a non-networked object

6.271.1 Detailed Description

Data structure storing spatial (Position/Rotation/Scale/Parent) synchronization data for spatial synchronization components, [NetworkTRSP](#) and its subclass [NetworkTransform](#).

6.271.2 Member Data Documentation

6.271.2.1 AreaOfInterestOverride

[NetworkId](#) [AreaOfInterestOverride](#)

Id of a behaviour used as the reference point for this component during area of interest operations The behaviour should be a [NetworkTRSP](#) derived class, that is on the same Transform as its associated [NetworkObject](#)

6.271.2.2 Parent

`NetworkBehaviourId` Parent

Id of a `NetworkBehaviour` on the parent of the component's transform.

6.271.2.3 Position

`Vector3` Position

Position relevant for the spatial synchronization component (can be used to either store a local position or a world position, depending on the component)

6.271.2.4 POSITION_OFFSET

```
const int POSITION_OFFSET = 2 [static]
```

Offset to point at the position values on the data buffer

6.271.2.5 Rotation

`Quaternion` Rotation

Rotation relevant for the spatial synchronization component (can be used to either store a local rotation or a world rotation, depending on the component)

6.271.2.6 Scale

`Vector3Compressed` Scale

Scale relevant for the spatial synchronization component

6.271.2.7 SIZE

```
const int SIZE = WORDS * Allocator.REPLICATE_WORD_SIZE [static]
```

The actual size for the networked properties in bytes

6.271.2.8 TeleportKey

```
int TeleportKey
```

Key used to differentiate between several teleports

6.271.2.9 WORDS

```
const int WORDS = 14 [static]
```

Networked properties word count for the base [NetworkTRSPData](#)

6.271.3 Property Documentation

6.271.3.1 NonNetworkedParent

```
NetworkBehaviourId NonNetworkedParent [static], [get]
```

Special [NetworkBehaviourId](#) value, used as a flag to tell the parent is a non-networked object

6.272 NormalizedRectAttribute Class Reference

Enables a special inspector drawer for Unity Rect type, specially designed for editing RectTransforms using normalized values.

Inherits PropertyAttribute.

Public Member Functions

- [NormalizedRectAttribute](#) (bool invertY=true, float aspectRatio=0)
Constructor for [NormalizedRectAttribute](#). InvertY inverts Y handling, for RectTransforms which treat lowerRight as origin, rather than upper left.

Public Attributes

- float [AspectRatio](#)
Set the Aspect Ratio
- bool [InvertY](#)
Signal if Y should be inverted

6.272.1 Detailed Description

Enables a special inspector drawer for Unity Rect type, specially designed for editing RectTransforms using normalized values.

6.272.2 Constructor & Destructor Documentation

6.272.2.1 NormalizedRectAttribute()

```
NormalizedRectAttribute (
    bool invertY = true,
    float aspectRatio = 0 )
```

Constructor for [NormalizedRectAttribute](#). InvertY inverts Y handling, for RectTransforms which treat lowerRight as origin, rather than upper left.

Parameters

<i>invertY</i>	Invert Y handling
<i>aspectRatio</i>	Expressed as Width/Height, this defines the ratio of the box shown in the inspector. Value of 0 indicates game window resolution will be used.

6.272.3 Member Data Documentation

6.272.3.1 AspectRatio

```
float AspectRatio
```

Set the Aspect Ratio

6.272.3.2 InvertY

```
bool InvertY
```

Signal if Y should be inverted

6.273 OnChangedRenderAttribute Class Reference

OnChangedRender Attribute

Inherits Attribute.

Public Member Functions

- [OnChangeRenderAttribute](#) (string methodName)
Initializes a new instance of the [OnChangeRenderAttribute](#) class.

Properties

- string [MethodName](#) [get]
Gets the name of the method to be called when the property changes.

6.273.1 Detailed Description

OnChangeRender Attribute

This attribute is used to specify a method that should be called when the property changes.

6.273.2 Constructor & Destructor Documentation

6.273.2.1 OnChangeRenderAttribute()

```
OnChangeRenderAttribute (  
    string methodName )
```

Initializes a new instance of the [OnChangeRenderAttribute](#) class.

Parameters

<i>methodName</i>	The name of the method to be called when the property changes.
-------------------	--

Exceptions

<i>ArgumentNullException</i>	Thrown when <i>methodName</i> is null or empty.
------------------------------	---

6.273.3 Property Documentation

6.273.3.1 MethodName

```
string MethodName [get]
```

Gets the name of the method to be called when the property changes.

6.274 PlayerRef Struct Reference

Represents a [Fusion](#) player.

Inherits [INetworkStruct](#), and `IEquatable< PlayerRef >`.

Public Member Functions

- override bool [Equals](#) (object obj)
Determines whether the specified object is equal to the current object.
- bool [Equals](#) ([PlayerRef](#) other)
Determines whether the specified [PlayerRef](#) is equal to the current [PlayerRef](#).
- override int [GetHashCode](#) ()
Serves as the default hash function.
- override string [ToString](#) ()
Returns a string that represents the current object.

Static Public Member Functions

- static [PlayerRef FromEncoded](#) (int encoded)
Creates a new [PlayerRef](#) from the given encoded value.
- static [PlayerRef FromIndex](#) (int index)
Creates a new [PlayerRef](#) from the given index.
- static bool [operator!=](#) ([PlayerRef](#) a, [PlayerRef](#) b)
Determines whether two [PlayerRef](#) instances are not equal.
- static bool [operator==](#) ([PlayerRef](#) a, [PlayerRef](#) b)
Determines whether two [PlayerRef](#) instances are equal.
- static unsafe [PlayerRef Read](#) ([NetBitBuffer](#) *buffer)
Reads a [PlayerRef](#) from the provided [NetBitBuffer](#).
- static unsafe void [Write](#) ([NetBitBuffer](#) *buffer, [PlayerRef](#) playerRef)
Writes the [PlayerRef](#) to the provided [NetBitBuffer](#).
- static unsafe void [Write](#)< T > (T *buffer, [PlayerRef](#) playerRef)
Writes the [PlayerRef](#) to the provided buffer.

Public Attributes

- int [_index](#)

Static Public Attributes

- const int [MASTER_CLIENT_RAW](#) = -1
A constant representing the raw index value for the master client.
- const int [SIZE](#) = 4
The size of the [PlayerRef](#) structure in bytes.

Properties

- int [AsIndex](#) [get]

Returns the *PlayerRef* int as an integer Id value.
- static [IEqualityComparer< PlayerRef > Comparer](#) = new [IndexEqualityComparer\(\)](#) [get]

Gets an equality comparer that can be used to compare two *PlayerRef* instances.
- static [PlayerRef Invalid](#) [get]

Invalid player ref. Used to differentiate no player ref (None) from an invalid one.
- bool [IsMasterClient](#) [get]

Returns true if this *PlayerRef* indicates the *MasterClient* rather than a specific *Player* by *Index*, This is a special flag value which has the encoded index value of -2 (internal raw backing value of -1). This is not a valid *PlayerRef* value in itself, and no *Runner* will ever be assigned this value as its *LocalPlayer*. It is used by properties like *Object.State* Authority to indicate that the *MasterClient* has authority (which ever player that currently is), rather than a specific *Player*.
- bool [IsNone](#) [get]

Returns true if the index value equals -1 (internal raw value of 0), indicating no player.
- bool [IsRealPlayer](#) [get]

If this player ref is a valid unique player index
- static [PlayerRef MasterClient](#) [get]

Special master client player ref value of -1
- static [PlayerRef None](#) [get]

None player
- int [PlayerId](#) [get]

Returns the *PlayerRef* as an integer Id value.
- int [RawEncoded](#) [get]

Returns the index backing value without modification. Unlike [AsIndex](#) which returns the backing value - 1.

6.274.1 Detailed Description

Represents a [Fusion](#) player.

The [PlayerRef](#), in contrast to the player index, is 1-based. The reason is that `default(PlayerRef)` will return a "null/invalid" player ref struct for convenience. There are automatic cast operators that can cast an int into a [PlayerRef](#).

```
default(PlayerRef), internally a 0, means NOBODY
PlayerRef, internally 1, is the same as player index 0
PlayerRef, internally 2, is the same as player index 1
```

6.274.2 Member Function Documentation

6.274.2.1 Equals() [1/2]

```
override bool Equals (
    object obj )
```

Determines whether the specified object is equal to the current object.

Parameters

<i>obj</i>	The object to compare with the current object.
------------	--

Returns

true if the specified object is equal to the current object; otherwise, false.

6.274.2.2 Equals() [2/2]

```
bool Equals (  
    PlayerRef other )
```

Determines whether the specified [PlayerRef](#) is equal to the current [PlayerRef](#).

Parameters

<i>other</i>	The PlayerRef to compare with the current PlayerRef .
--------------	---

Returns

true if the specified [PlayerRef](#) is equal to the current [PlayerRef](#); otherwise, false.

6.274.2.3 FromEncoded()

```
static PlayerRef FromEncoded (  
    int encoded ) [static]
```

Creates a new [PlayerRef](#) from the given encoded value.

Parameters

<i>encoded</i>	The encoded value to create the PlayerRef from.
----------------	---

Returns

A new [PlayerRef](#) that represents the encoded value.

6.274.2.4 FromIndex()

```
static PlayerRef FromIndex (  
    int index ) [static]
```

Creates a new [PlayerRef](#) from the given index.

Parameters

<i>index</i>	The index to create the PlayerRef from.
--------------	---

Returns

A new [PlayerRef](#) that represents the index.

6.274.2.5 GetHashCode()

```
override int GetHashCode ( )
```

Serves as the default hash function.

6.274.2.6 operator"!="()

```
static bool operator!= (
    PlayerRef a,
    PlayerRef b ) [static]
```

Determines whether two [PlayerRef](#) instances are not equal.

Parameters

<i>a</i>	The first PlayerRef to compare.
<i>b</i>	The second PlayerRef to compare.

Returns

true if the [PlayerRefs](#) are not equal; otherwise, false.

6.274.2.7 operator=="()

```
static bool operator== (
    PlayerRef a,
    PlayerRef b ) [static]
```

Determines whether two [PlayerRef](#) instances are equal.

Parameters

<i>a</i>	The first PlayerRef to compare.
<i>b</i>	The second PlayerRef to compare.

Returns

true if the PlayerRefs are equal; otherwise, false.

6.274.2.8 Read()

```
static unsafe PlayerRef Read (
    NetBitBuffer * buffer ) [static]
```

Reads a [PlayerRef](#) from the provided NetBitBuffer.

Parameters

<i>buffer</i>	The buffer to read from.
---------------	--------------------------

Returns

The [PlayerRef](#) read from the buffer.

6.274.2.9 ToString()

```
override string ToString ( )
```

Returns a string that represents the current object.

6.274.2.10 Write()

```
static unsafe void Write (
    NetBitBuffer * buffer,
    PlayerRef playerRef ) [static]
```

Writes the [PlayerRef](#) to the provided NetBitBuffer.

Parameters

<i>buffer</i>	The buffer to write to.
<i>playerRef</i>	The PlayerRef to write.

6.274.2.11 Write< T >()

```
static unsafe void Write< T > (
```

```
T * buffer,
PlayerRef playerRef ) [static]
```

Writes the [PlayerRef](#) to the provided buffer.

Template Parameters

<i>T</i>	The type of the buffer. Must be unmanaged and implement INetBitWriteStream .
----------	--

Parameters

<i>buffer</i>	The buffer to write to.
<i>playerRef</i>	The PlayerRef to write.

Type Constraints

T : *unmanaged*

T : *INetBitWriteStream*

6.274.3 Member Data Documentation

6.274.3.1 MASTER_CLIENT_RAW

```
const int MASTER_CLIENT_RAW = -1 [static]
```

A constant representing the raw index value for the master client.

6.274.3.2 SIZE

```
const int SIZE = 4 [static]
```

The size of the [PlayerRef](#) structure in bytes.

6.274.4 Property Documentation

6.274.4.1 AsIndex

```
int AsIndex [get]
```

Returns the [PlayerRef](#) int as an integer Id value.

-1=None -2=MasterClient >=0=PlayerId

6.274.4.2 Comparer

```
IEqualityComparer<PlayerRef> Comparer = new IndexEqualityComparer() [static], [get]
```

Gets an equality comparer that can be used to compare two [PlayerRef](#) instances.

6.274.4.3 Invalid

```
PlayerRef Invalid [static], [get]
```

Invalid player ref. Used to differentiate no player ref (None) from an invalid one.

6.274.4.4 IsMasterClient

```
bool IsMasterClient [get]
```

Returns true if this [PlayerRef](#) indicates the MasterClient rather than a specific Player by Index, This is a special flag value which has the encoded index value of -2 (internal raw backing value of -1). This is not a valid [PlayerRef](#) value in itself, and no Runner will ever be assigned this value as its LocalPlayer. It is used by properties like `Object.StateAuthority` to indicate that the MasterClient has authority (which ever player that currently is), rather than a specific Player.

6.274.4.5 IsNone

```
bool IsNone [get]
```

Returns true if the index value equals -1 (internal raw value of 0), indicating no player.

6.274.4.6 IsRealPlayer

```
bool IsRealPlayer [get]
```

If this player ref is a valid unique player index

6.274.4.7 MasterClient

```
PlayerRef MasterClient [static], [get]
```

Special master client player ref value of -1

6.274.4.8 None

`PlayerRef` None [static], [get]

None player

6.274.4.9 PlayerId

`int` PlayerId [get]

Returns the `PlayerRef` as an integer Id value.

-1=None -2=MasterClient

6.274.4.10 RawEncoded

`int` RawEncoded [get]

Returns the index backing value without modification. Unlike `AsIndex` which returns the backing value - 1.

0=None -1=MasterClient >0=PlayerId

6.275 PreserveInPluginAttribute Class Reference

Preserve In Plugin Attribute

Inherits Attribute.

Public Member Functions

- `PreserveInPluginAttribute` ()

Properties

- bool `KeepNonStateMembers` = true [get, set]

6.275.1 Detailed Description

Preserve In Plugin Attribute

6.275.2 Constructor & Destructor Documentation

6.275.2.1 PreserveInPluginAttribute()

```
PreserveInPluginAttribute ( )
```

6.275.3 Property Documentation

6.275.3.1 KeepNonStateMembers

```
bool KeepNonStateMembers = true [get], [set]
```

6.276 Primes Class Reference

Provides a set of methods to work with prime numbers.

Static Public Member Functions

- static int [GetNextPrime](#) (int value)
Gets the next prime number greater than the specified value.
- static uint [GetNextPrime](#) (uint value)
Gets the next prime number greater than the specified unsigned value.
- static bool [IsPrime](#) (int value)
Determines whether the specified value is a prime number.

6.276.1 Detailed Description

Provides a set of methods to work with prime numbers.

6.276.2 Member Function Documentation

6.276.2.1 GetNextPrime() [1/2]

```
static int GetNextPrime (  
    int value ) [static]
```

Gets the next prime number greater than the specified value.

Parameters

<i>value</i>	The value to find the next prime number for.
--------------	--

Returns

The next prime number greater than the specified value.

Exceptions

<i>InvalidOperationException</i>	Thrown when there is no larger prime number in the table.
----------------------------------	---

6.276.2.2 GetNextPrime() [2/2]

```
static uint GetNextPrime (  
    uint value ) [static]
```

Gets the next prime number greater than the specified unsigned value.

Parameters

<i>value</i>	The unsigned value to find the next prime number for.
--------------	---

Returns

The next prime number greater than the specified unsigned value.

Exceptions

<i>InvalidOperationException</i>	Thrown when there is no larger prime number in the table.
----------------------------------	---

6.276.2.3 IsPrime()

```
static bool IsPrime (  
    int value ) [static]
```

Determines whether the specified value is a prime number.

Parameters

<i>value</i>	The value to check for primality.
--------------	-----------------------------------

Returns

`true` if the specified value is a prime number; otherwise, `false`.

6.277 PropertyAttribute Class Reference

Specifies that the attribute can be applied to fields only.

Inherits `PropertyAttribute`.

Inherited by [DecoratingPropertyAttribute](#), [DefaultForPropertyAttribute](#), [DrawerPropertyAttribute](#), [FixedBufferPropertyAttribute](#), [NetworkPrefabAttribute](#), [ResolveNetworkPrefabSourceAttribute](#), [ScriptHelpAttribute](#), [SerializableTypeAttribute](#), and [UnityAddressablesRuntimeKeyAttribute](#).

6.277.1 Detailed Description

Specifies that the attribute can be applied to fields only.

6.278 BitStream Class Reference

[BitStream](#) serialization methods.

Public Member Functions

- delegate void [ArrayElementSerializer< T >](#) (ref T element)
Serialize a value.
- [BitStream](#) ()
Initializes a new instance of the [BitStream](#) class with an empty byte array.
- [BitStream](#) (byte[] arr)
Initializes a new instance of the [BitStream](#) class with the specified byte array.
- [BitStream](#) (byte[] arr, int size)
Initializes a new instance of the [BitStream](#) class with the specified byte array and size.
- [BitStream](#) (int size)
Initializes a new instance of the [BitStream](#) class with a specified size in bytes.
- bool [CanRead](#) ()
Checks if at least one bit can be read from the stream.
- bool [CanRead](#) (int bits)
Checks if the specified number of bits can be read from the stream.
- bool [CanWrite](#) ()
Checks if at least one bit can be written to the stream.
- bool [CanWrite](#) (int bits)
Checks if the specified number of bits can be written to the stream.
- bool [Condition](#) (bool condition)
Evaluates a condition and serializes it if in write mode, or deserializes it if in read mode.
- void [CopyFromArray](#) (Byte[] array)
Copies data from the specified byte array into the internal buffer.
- void [Expand](#) ()

- Doubles the capacity of the internal buffer.*
- bool [ReadBool](#) ()
Reads a boolean value from the stream.
- bool [ReadBoolean](#) ()
Reads a boolean value from the stream.
- byte [ReadByte](#) ()
Reads a byte value from the stream.
- byte [ReadByte](#) (int bits)
Reads a byte value from the stream with a specified number of bits.
- void [ReadByteArray](#) (byte[] to)
Reads bytes from the stream into the specified byte array.
- void [ReadByteArray](#) (byte[] to, int count)
Reads a specified number of bytes from the stream into the specified byte array.
- void [ReadByteArray](#) (byte[] to, int offset, int count)
Reads a specified number of bytes from the stream into the specified byte array starting at a given offset.
- byte[] [ReadByteArray](#) (int size)
Reads a specified number of bytes from the stream into a new byte array.
- byte[] [ReadByteArrayLengthPrefixed](#) ()
Reads a byte array from the stream with a length prefix.
- Char [ReadChar](#) ()
Reads a character value from the stream.
- double [ReadDouble](#) ()
Reads a 64-bit floating point value from the stream.
- float [ReadFloat](#) ()
Reads a 32-bit floating point value from the stream.
- Guid [ReadGuid](#) ()
Reads a GUID from the stream.
- int [ReadInt](#) ()
Reads a 32-bit integer value from the stream.
- int [ReadInt](#) (int bits)
Reads a 32-bit integer value from the stream with a specified number of bits.
- int [ReadInt_Shifted](#) (int bits, int shift)
Reads a shifted 32-bit integer value from the stream with a specified number of bits.
- long [ReadLong](#) ()
Reads a signed 64-bit integer value from the stream.
- long [ReadLong](#) (int bits)
Reads a signed 64-bit integer value from the stream with a specified number of bits.
- sbyte [ReadSByte](#) ()
Reads a signed byte value from the stream.
- short [ReadShort](#) ()
Reads a short value from the stream.
- short [ReadShort](#) (int bits)
Reads a short value from the stream with a specified number of bits.
- string [ReadString](#) ()
Reads a string from the stream using UTF-8 encoding.
- string [ReadString](#) (Encoding encoding)
Reads a string from the stream using the specified encoding.
- uint [ReadUInt](#) ()
Reads an unsigned 32-bit integer value from the stream.
- uint [ReadUInt](#) (int bits)
Reads an unsigned 32-bit integer value from the stream with a specified number of bits.

- `ulong ReadULong ()`
Reads an unsigned 64-bit integer value from the stream.
- `ulong ReadULong (int bits)`
Reads an unsigned 64-bit integer value from the stream with a specified number of bits.
- `ushort ReadUShort ()`
Reads an unsigned short value from the stream.
- `ushort ReadUShort (int bits)`
Reads an unsigned short value from the stream with a specified number of bits.
- `void Reset ()`
Resets the internal buffer to its initial state.
- `void Reset (Int32 byteSize)`
Resets the internal buffer to the specified size.
- `void ResetFast (Int32 byteSize)`
Resets the internal buffer to the specified size without clearing the data.
- `int RoundToByte ()`
Rounds the current position to the nearest byte boundary by filling with zero bits if necessary.
- `unsafe void Serialize (byte *v)`
Serializes or deserializes a byte value.
- `unsafe void Serialize (int *v)`
Serializes or deserializes an integer value.
- `unsafe void Serialize (int *v, int bits)`
Serializes or deserializes a 32-bit integer value with a specified number of bits.
- `unsafe void Serialize (long *v)`
Serializes or deserializes a 64-bit integer value.
- `void Serialize (ref bool value)`
Serializes or deserializes a boolean value.
- `void Serialize (ref Byte value)`
Serializes or deserializes a byte value.
- `void Serialize (ref Byte[] array, ref Int32 length)`
Serializes or deserializes a byte array with a specified length.
- `void Serialize (ref Byte[] array, ref Int32 length, Int32 fixedSize)`
Serializes or deserializes a byte array with a specified length and fixed size.
- `void Serialize (ref Byte[] value)`
Serializes or deserializes a byte array with a length prefix.
- `void Serialize (ref Byte[] value, Int32 fixedSize)`
Serializes or deserializes a byte array with a fixed size.
- `void Serialize (ref Double value)`
Serializes or deserializes a double value.
- `void Serialize (ref float value)`
Serializes or deserializes a float value.
- `void Serialize (ref int value)`
Serializes or deserializes a 32-bit integer value.
- `void Serialize (ref int value, int bits)`
Serializes or deserializes a 32-bit integer value with a specified number of bits.
- `void Serialize (ref Int32[] value)`
Serializes or deserializes an array of 32-bit integers.
- `void Serialize (ref Int64 value)`
Serializes or deserializes a 64-bit integer value.
- `void Serialize (ref String value)`
Serializes or deserializes a string value.
- `void Serialize (ref uint value)`

- Serializes or deserializes a 32-bit unsigned integer value.*

 - void [Serialize](#) (ref uint value, int bits)
- Serializes or deserializes a 32-bit unsigned integer value with a specified number of bits.*

 - void [Serialize](#) (ref UInt64 value)
- Serializes or deserializes a 64-bit unsigned integer value.*

 - void [Serialize](#) (ref ulong value, int bits)
- Serializes or deserializes a 64-bit unsigned integer value with a specified number of bits.*

 - unsafe void [Serialize](#) (sbyte *v)
- Serializes or deserializes a signed byte value.*

 - unsafe void [Serialize](#) (short *v)
- Serializes or deserializes a short value.*

 - unsafe void [Serialize](#) (uint *v)
- Serializes or deserializes an unsigned 32-bit integer value.*

 - unsafe void [Serialize](#) (uint *v, int bits)
- Serializes or deserializes an unsigned 32-bit integer value with a specified number of bits.*

 - unsafe void [Serialize](#) (ulong *v)
- Serializes or deserializes an unsigned 64-bit integer value.*

 - unsafe void [Serialize](#) (ushort *v)
- Serializes or deserializes an unsigned short value.*

 - void [SerializeArray](#)< T > (ref T[] array, ArrayElementSerializer< T > serializer)
- Serializes or deserializes an array of elements using a specified serializer.*

 - void [SerializeArrayLength](#)< T > (ref T[] array)
- Serializes or deserializes the length of an array.*

 - unsafe void [SerializeBuffer](#) (byte *buffer, int length)
- Serializes or deserializes a buffer of byte values.*

 - unsafe void [SerializeBuffer](#) (int *buffer, int length)
- Serializes or deserializes a buffer of 32-bit integer values.*

 - unsafe void [SerializeBuffer](#) (long *buffer, int length)
- Serializes or deserializes a buffer of 64-bit integer values.*

 - unsafe void [SerializeBuffer](#) (sbyte *buffer, int length)
- Serializes or deserializes a buffer of signed byte values.*

 - unsafe void [SerializeBuffer](#) (short *buffer, int length)
- Serializes or deserializes a buffer of short values.*

 - unsafe void [SerializeBuffer](#) (uint *buffer, int length)
- Serializes or deserializes a buffer of unsigned 32-bit integer values.*

 - unsafe void [SerializeBuffer](#) (ulong *buffer, int length)
- Serializes or deserializes a buffer of unsigned 64-bit integer values.*

 - unsafe void [SerializeBuffer](#) (ushort *buffer, int length)
- Serializes or deserializes a buffer of unsigned short values.*

 - void [SetBuffer](#) (byte[] arr)
- Sets the internal buffer to the specified byte array.*

 - void [SetBuffer](#) (byte[] arr, int size)
- Sets the internal buffer to the specified byte array and size.*

 - byte[] [ToArray](#) ()
- Converts the internal buffer to a byte array.*

 - bool [WriteBool](#) (bool value)
- Writes a boolean value to the stream.*

 - bool [WriteBoolean](#) (bool value)
- Writes a boolean value to the stream.*

 - void [WriteByte](#) (byte value)
- Writes a byte value to the stream.*

- void [WriteByte](#) (byte value, int bits)
Writes a byte value to the stream with a specified number of bits.
- void [WriteByteArray](#) (byte[] from)
Writes a byte array to the stream.
- void [WriteByteArray](#) (byte[] from, int count)
Writes a specified number of bytes from a byte array to the stream.
- void [WriteByteArray](#) (byte[] from, int offset, int count)
Writes a specified number of bytes from a byte array to the stream starting at a given offset.
- void [WriteByteArrayLengthPrefixed](#) (byte[] array)
Writes a byte array to the stream with a length prefix.
- void [WriteByteArrayLengthPrefixed](#) (byte[] array, int maxLength)
Writes a byte array to the stream with a length prefix and a specified maximum length.
- void [WriteChar](#) (Char value)
Writes a character value to the stream.
- void [WriteDouble](#) (double value)
Writes a 64-bit floating point value to the stream.
- void [WriteFloat](#) (float value)
Writes a 32-bit floating point value to the stream.
- void [WriteGuid](#) (Guid guid)
Writes a GUID to the stream.
- void [WriteInt](#) (int value)
Writes a 32-bit integer value to the stream.
- void [WriteInt](#) (int value, int bits)
Writes a 32-bit integer value to the stream with a specified number of bits.
- void [WriteInt_Shifted](#) (int value, int bits, int shift)
Writes a shifted 32-bit integer value to the stream with a specified number of bits.
- void [WriteLong](#) (long value)
Writes a signed 64-bit integer value to the stream.
- void [WriteLong](#) (long value, int bits)
Writes a signed 64-bit integer value to the stream with a specified number of bits.
- void [WriteSByte](#) (sbyte value)
Writes a signed byte value to the stream.
- void [WriteShort](#) (short value)
Writes a short value to the stream.
- void [WriteShort](#) (short value, int bits)
Writes a short value to the stream with a specified number of bits.
- void [WriteString](#) (string value)
Writes a string to the stream using UTF-8 encoding.
- void [WriteString](#) (string value, Encoding encoding)
Writes a string to the stream using the specified encoding.
- void [WriteUInt](#) (uint value)
Writes an unsigned 32-bit integer value to the stream.
- void [WriteUInt](#) (uint value, int bits)
Writes an unsigned 32-bit integer value to the stream with a specified number of bits.
- void [WriteULong](#) (ulong value)
Writes an unsigned 64-bit integer value to the stream.
- void [WriteULong](#) (ulong value, int bits)
Writes an unsigned 64-bit integer value to the stream with a specified number of bits.
- void [WriteUShort](#) (ushort value)
Writes an unsigned short value to the stream.
- void [WriteUShort](#) (ushort value, int bits)
Writes an unsigned short value to the stream with a specified number of bits.

Static Public Member Functions

- static void [WriteByteAt](#) (byte[] data, int ptr, int bits, byte value)
Writes a byte value at a specified bit position in a byte array.

Properties

- int [BytesRequired](#) [get]
Size of written buffer in BYTES Ammount of bytes required considering the total of written bytes
- int [Capacity](#) [get]
Total Size in BYTES of the Buffer
- Byte[] [Data](#) [get]
Internal Byte Array
- bool [Done](#) [get]
Signal if the buffer was completely written
- bool [IsEvenBytes](#) [get]
Gets a value indicating whether the current position is at an even byte boundary.
- bool [Overflowing](#) [get]
Signal if the buffer is overflowing
- int [Position](#) [get, set]
Current read/write position in BITS inside the Buffer
- bool [Reading](#) [get, set]
Signal if the Buffer is in Read Mode
- int [Size](#) [get, set]
Total size in BITS of the buffer
- bool [Writing](#) [get, set]
Signal if the Buffer is in Write Mode

6.278.1 Detailed Description

[BitStream](#) serialization methods.

6.278.2 Constructor & Destructor Documentation

6.278.2.1 [BitStream\(\)](#) [1/4]

```
BitStream ( )
```

Initializes a new instance of the [BitStream](#) class with an empty byte array.

6.278.2.2 [BitStream\(\)](#) [2/4]

```
BitStream (
    Int32 size )
```

Initializes a new instance of the [BitStream](#) class with a specified size in bytes.

Parameters

<i>size</i>	The size of the byte array in bytes.
-------------	--------------------------------------

6.278.2.3 BitStream() [3/4]

```
BitStream (
    byte[] arr )
```

Initializes a new instance of the [BitStream](#) class with the specified byte array.

Parameters

<i>arr</i>	The byte array to use as the internal buffer.
------------	---

6.278.2.4 BitStream() [4/4]

```
BitStream (
    byte[] arr,
    int size )
```

Initializes a new instance of the [BitStream](#) class with the specified byte array and size.

Parameters

<i>arr</i>	The byte array to use as the internal buffer.
<i>size</i>	The size of the byte array in bytes.

6.278.3 Member Function Documentation**6.278.3.1 ArrayElementSerializer< T >()**

```
delegate void ArrayElementSerializer< T > (
    ref T element )
```

Serialize a value.

Template Parameters

<i>T</i>	The value type.
----------	-----------------

6.278.3.2 CanRead() [1/2]

```
bool CanRead ( )
```

Checks if at least one bit can be read from the stream.

Returns

True if at least one bit can be read, otherwise false.

6.278.3.3 CanRead() [2/2]

```
bool CanRead (
    int bits )
```

Checks if the specified number of bits can be read from the stream.

Parameters

<i>bits</i>	The number of bits to check.
-------------	------------------------------

Returns

True if the specified number of bits can be read, otherwise false.

6.278.3.4 CanWrite() [1/2]

```
bool CanWrite ( )
```

Checks if at least one bit can be written to the stream.

Returns

True if at least one bit can be written, otherwise false.

6.278.3.5 CanWrite() [2/2]

```
bool CanWrite (
    int bits )
```

Checks if the specified number of bits can be written to the stream.

Parameters

<i>bits</i>	The number of bits to check.
-------------	------------------------------

Returns

True if the specified number of bits can be written, otherwise false.

6.278.3.6 Condition()

```
bool Condition (  
    bool condition )
```

Evaluates a condition and serializes it if in write mode, or deserializes it if in read mode.

Parameters

<i>condition</i>	The condition to evaluate and serialize/deserialize.
------------------	--

Returns

The evaluated condition.

6.278.3.7 CopyFromArray()

```
void CopyFromArray (  
    Byte[] array )
```

Copies data from the specified byte array into the internal buffer.

Parameters

<i>array</i>	The byte array to copy from.
--------------	------------------------------

6.278.3.8 Expand()

```
void Expand ( )
```

Doubles the capacity of the internal buffer.

6.278.3.9 ReadBool()

```
bool ReadBool ( )
```

Reads a boolean value from the stream.

Returns

The boolean value that was read.

6.278.3.10 ReadBoolean()

```
bool ReadBoolean ( )
```

Reads a boolean value from the stream.

Returns

The boolean value that was read.

6.278.3.11 ReadByte() [1/2]

```
byte ReadByte ( )
```

Reads a byte value from the stream.

Returns

The byte value that was read.

6.278.3.12 ReadByte() [2/2]

```
byte ReadByte (
    int bits )
```

Reads a byte value from the stream with a specified number of bits.

Parameters

<i>bits</i>	The number of bits to read.
-------------	-----------------------------

Returns

The byte value that was read.

6.278.3.13 ReadByteArray() [1/4]

```
void ReadByteArray (
    byte[] to )
```

Reads bytes from the stream into the specified byte array.

Parameters

<i>to</i>	The byte array to read into.
-----------	------------------------------

6.278.3.14 ReadByteArray() [2/4]

```
void ReadByteArray (
    byte[] to,
    int count )
```

Reads a specified number of bytes from the stream into the specified byte array.

Parameters

<i>to</i>	The byte array to read into.
<i>count</i>	The number of bytes to read.

6.278.3.15 ReadByteArray() [3/4]

```
void ReadByteArray (
    byte[] to,
    int offset,
    int count )
```

Reads a specified number of bytes from the stream into the specified byte array starting at a given offset.

Parameters

<i>to</i>	The byte array to read into.
<i>offset</i>	The starting offset in the byte array.
<i>count</i>	The number of bytes to read.

6.278.3.16 ReadByteArray() [4/4]

```
byte [] ReadByteArray (
    int size )
```

Reads a specified number of bytes from the stream into a new byte array.

Parameters

<i>size</i>	The number of bytes to read.
-------------	------------------------------

Returns

A byte array containing the read bytes.

6.278.3.17 ReadByteArrayLengthPrefixed()

```
byte [] ReadByteArrayLengthPrefixed ( )
```

Reads a byte array from the stream with a length prefix.

Returns

A byte array containing the read bytes, or null if the length prefix indicates no data.

6.278.3.18 ReadChar()

```
Char ReadChar ( )
```

Reads a character value from the stream.

Returns

The character value that was read.

6.278.3.19 ReadDouble()

```
double ReadDouble ( )
```

Reads a 64-bit floating point value from the stream.

Returns

The 64-bit floating point value that was read.

6.278.3.20 ReadFloat()

```
float ReadFloat ( )
```

Reads a 32-bit floating point value from the stream.

Returns

The 32-bit floating point value that was read.

6.278.3.21 ReadGuid()

```
Guid ReadGuid ( )
```

Reads a GUID from the stream.

Returns

The GUID that was read.

6.278.3.22 ReadInt() [1/2]

```
int ReadInt ( )
```

Reads a 32-bit integer value from the stream.

Returns

The 32-bit integer value that was read.

6.278.3.23 ReadInt() [2/2]

```
int ReadInt (
    int bits )
```

Reads a 32-bit integer value from the stream with a specified number of bits.

Parameters

<i>bits</i>	The number of bits to read.
-------------	-----------------------------

Returns

The 32-bit integer value that was read.

6.278.3.24 ReadInt_Shifted()

```
int ReadInt_Shifted (
    int bits,
    int shift )
```

Reads a shifted 32-bit integer value from the stream with a specified number of bits.

Parameters

<i>bits</i>	The number of bits to read.
<i>shift</i>	The number of bits to shift.

Returns

The 32-bit integer value that was read.

6.278.3.25 ReadLong() [1/2]

```
long ReadLong ( )
```

Reads a signed 64-bit integer value from the stream.

Returns

The signed 64-bit integer value that was read.

6.278.3.26 ReadLong() [2/2]

```
long ReadLong (
    int bits )
```

Reads a signed 64-bit integer value from the stream with a specified number of bits.

Parameters

<i>bits</i>	The number of bits to read.
-------------	-----------------------------

Returns

The signed 64-bit integer value that was read.

6.278.3.27 ReadSByte()

```
sbyte ReadSByte ( )
```

Reads a signed byte value from the stream.

Returns

The signed byte value that was read.

6.278.3.28 ReadShort() [1/2]

```
short ReadShort ( )
```

Reads a short value from the stream.

Returns

The short value that was read.

6.278.3.29 ReadShort() [2/2]

```
short ReadShort (
    int bits )
```

Reads a short value from the stream with a specified number of bits.

Parameters

<i>bits</i>	The number of bits to read.
-------------	-----------------------------

Returns

The short value that was read.

6.278.3.30 ReadString() [1/2]

```
string ReadString ( )
```

Reads a string from the stream using UTF-8 encoding.

Returns

The string that was read, or null if the length prefix indicates no data.

6.278.3.31 ReadString() [2/2]

```
string ReadString (
    Encoding encoding )
```

Reads a string from the stream using the specified encoding.

Parameters

<i>encoding</i>	The encoding to use for the string.
-----------------	-------------------------------------

Returns

The string that was read, or null if the length prefix indicates no data.

6.278.3.32 ReadUInt() [1/2]

```
uint ReadUInt ( )
```

Reads an unsigned 32-bit integer value from the stream.

Returns

The unsigned 32-bit integer value that was read.

6.278.3.33 ReadUInt() [2/2]

```
uint ReadUInt (
    int bits )
```

Reads an unsigned 32-bit integer value from the stream with a specified number of bits.

Parameters

<i>bits</i>	The number of bits to read.
-------------	-----------------------------

Returns

The unsigned 32-bit integer value that was read.

6.278.3.34 ReadULong() [1/2]

```
ulong ReadULong ( )
```

Reads an unsigned 64-bit integer value from the stream.

Returns

The unsigned 64-bit integer value that was read.

6.278.3.35 ReadULong() [2/2]

```
ulong ReadULong (
    int bits )
```

Reads an unsigned 64-bit integer value from the stream with a specified number of bits.

Parameters

<i>bits</i>	The number of bits to read.
-------------	-----------------------------

Returns

The unsigned 64-bit integer value that was read.

6.278.3.36 ReadUShort() [1/2]

```
ushort ReadUShort ( )
```

Reads an unsigned short value from the stream.

Returns

The unsigned short value that was read.

6.278.3.37 ReadUShort() [2/2]

```
ushort ReadUShort (
    int bits )
```

Reads an unsigned short value from the stream with a specified number of bits.

Parameters

<i>bits</i>	The number of bits to read.
-------------	-----------------------------

Returns

The unsigned short value that was read.

6.278.3.38 Reset() [1/2]

```
void Reset ( )
```

Resets the internal buffer to its initial state.

6.278.3.39 Reset() [2/2]

```
void Reset (
    Int32 byteSize )
```

Resets the internal buffer to the specified size.

Parameters

<i>byteSize</i>	The size in bytes to reset the buffer to.
-----------------	---

6.278.3.40 ResetFast()

```
void ResetFast (
    Int32 byteSize )
```

Resets the internal buffer to the specified size without clearing the data.

Parameters

<i>byteSize</i>	The size in bytes to reset the buffer to.
-----------------	---

6.278.3.41 RoundToByte()

```
int RoundToByte ( )
```

Rounds the current position to the nearest byte boundary by filling with zero bits if necessary.

Returns

The number of bytes required to store the current position.

6.278.3.42 Serialize() [1/27]

```
unsafe void Serialize (
    byte * v )
```

Serializes or deserializes a byte value.

Parameters

<code>v</code>	The byte value to serialize/deserialize.
----------------	--

6.278.3.43 Serialize() [2/27]

```
unsafe void Serialize (
    int * v )
```

Serializes or deserializes an integer value.

Parameters

<code>v</code>	The integer value to serialize/deserialize.
----------------	---

6.278.3.44 Serialize() [3/27]

```
unsafe void Serialize (
    int * v,
    int bits )
```

Serializes or deserializes a 32-bit integer value with a specified number of bits.

Parameters

<i>v</i>	The 32-bit integer value to serialize/deserialize.
<i>bits</i>	The number of bits to use for serialization/deserialization.

6.278.3.45 Serialize() [4/27]

```
unsafe void Serialize (  
    long * v )
```

Serializes or deserializes a 64-bit integer value.

Parameters

<i>v</i>	The 64-bit integer value to serialize/deserialize.
----------	--

6.278.3.46 Serialize() [5/27]

```
void Serialize (  
    ref bool value )
```

Serializes or deserializes a boolean value.

Parameters

<i>value</i>	The boolean value to serialize/deserialize.
--------------	---

6.278.3.47 Serialize() [6/27]

```
void Serialize (  
    ref Byte value )
```

Serializes or deserializes a byte value.

Parameters

<i>value</i>	The byte value to serialize/deserialize.
--------------	--

6.278.3.48 Serialize() [7/27]

```
void Serialize (
    ref Byte[] array,
    ref Int32 length )
```

Serializes or deserializes a byte array with a specified length.

Parameters

<i>array</i>	The byte array to serialize/deserialize.
<i>length</i>	The length of the byte array.

6.278.3.49 Serialize() [8/27]

```
void Serialize (
    ref Byte[] array,
    ref Int32 length,
    Int32 fixedSize )
```

Serializes or deserializes a byte array with a specified length and fixed size.

Parameters

<i>array</i>	The byte array to serialize/deserialize.
<i>length</i>	The length of the byte array.
<i>fixedSize</i>	The fixed size of the byte array.

6.278.3.50 Serialize() [9/27]

```
void Serialize (
    ref Byte[] value )
```

Serializes or deserializes a byte array with a length prefix.

Parameters

<i>value</i>	The byte array to serialize/deserialize.
--------------	--

6.278.3.51 Serialize() [10/27]

```
void Serialize (
```

```
ref Byte[] value,  
Int32 fixedSize )
```

Serializes or deserializes a byte array with a fixed size.

Parameters

<i>value</i>	The byte array to serialize/deserialize.
<i>fixedSize</i>	The fixed size of the byte array.

6.278.3.52 Serialize() [11/27]

```
void Serialize (  
    ref Double value )
```

Serializes or deserializes a double value.

Parameters

<i>value</i>	The double value to serialize/deserialize.
--------------	--

6.278.3.53 Serialize() [12/27]

```
void Serialize (  
    ref float value )
```

Serializes or deserializes a float value.

Parameters

<i>value</i>	The float value to serialize/deserialize.
--------------	---

6.278.3.54 Serialize() [13/27]

```
void Serialize (  
    ref int value )
```

Serializes or deserializes a 32-bit integer value.

Parameters

<i>value</i>	The 32-bit integer value to serialize/deserialize.
--------------	--

6.278.3.55 Serialize() [14/27]

```
void Serialize (
    ref int value,
    int bits )
```

Serializes or deserializes a 32-bit integer value with a specified number of bits.

Parameters

<i>value</i>	The 32-bit integer value to serialize/deserialize.
<i>bits</i>	The number of bits to use for serialization/deserialization.

6.278.3.56 Serialize() [15/27]

```
void Serialize (
    ref Int32[] value )
```

Serializes or deserializes an array of 32-bit integers.

Parameters

<i>value</i>	The array of 32-bit integers to serialize/deserialize.
--------------	--

6.278.3.57 Serialize() [16/27]

```
void Serialize (
    ref Int64 value )
```

Serializes or deserializes a 64-bit integer value.

Parameters

<i>value</i>	The 64-bit integer value to serialize/deserialize.
--------------	--

6.278.3.58 Serialize() [17/27]

```
void Serialize (
    ref String value )
```

Serializes or deserializes a string value.

Parameters

<i>value</i>	The string value to serialize/deserialize.
--------------	--

6.278.3.59 Serialize() [18/27]

```
void Serialize (  
    ref uint value )
```

Serializes or deserializes a 32-bit unsigned integer value.

Parameters

<i>value</i>	The 32-bit unsigned integer value to serialize/deserialize.
--------------	---

6.278.3.60 Serialize() [19/27]

```
void Serialize (  
    ref uint value,  
    int bits )
```

Serializes or deserializes a 32-bit unsigned integer value with a specified number of bits.

Parameters

<i>value</i>	The 32-bit unsigned integer value to serialize/deserialize.
<i>bits</i>	The number of bits to use for serialization/deserialization.

6.278.3.61 Serialize() [20/27]

```
void Serialize (  
    ref UInt64 value )
```

Serializes or deserializes a 64-bit unsigned integer value.

Parameters

<i>value</i>	The 64-bit unsigned integer value to serialize/deserialize.
--------------	---

6.278.3.62 Serialize() [21/27]

```
void Serialize (
    ref ulong value,
    int bits )
```

Serializes or deserializes a 64-bit unsigned integer value with a specified number of bits.

Parameters

<i>value</i>	The 64-bit unsigned integer value to serialize/deserialize.
<i>bits</i>	The number of bits to use for serialization/deserialization.

6.278.3.63 Serialize() [22/27]

```
unsafe void Serialize (
    sbyte * v )
```

Serializes or deserializes a signed byte value.

Parameters

<i>v</i>	The signed byte value to serialize/deserialize.
----------	---

6.278.3.64 Serialize() [23/27]

```
unsafe void Serialize (
    short * v )
```

Serializes or deserializes a short value.

Parameters

<i>v</i>	The short value to serialize/deserialize.
----------	---

6.278.3.65 Serialize() [24/27]

```
unsafe void Serialize (
    uint * v )
```

Serializes or deserializes an unsigned 32-bit integer value.

Parameters

<i>v</i>	The unsigned 32-bit integer value to serialize/deserialize.
----------	---

6.278.3.66 Serialize() [25/27]

```
unsafe void Serialize (  
    uint * v,  
    int bits )
```

Serializes or deserializes an unsigned 32-bit integer value with a specified number of bits.

Parameters

<i>v</i>	The unsigned 32-bit integer value to serialize/deserialize.
<i>bits</i>	The number of bits to use for serialization/deserialization.

6.278.3.67 Serialize() [26/27]

```
unsafe void Serialize (  
    ulong * v )
```

Serializes or deserializes an unsigned 64-bit integer value.

Parameters

<i>v</i>	The unsigned 64-bit integer value to serialize/deserialize.
----------	---

6.278.3.68 Serialize() [27/27]

```
unsafe void Serialize (  
    ushort * v )
```

Serializes or deserializes an unsigned short value.

Parameters

<i>v</i>	The unsigned short value to serialize/deserialize.
----------	--

6.278.3.69 SerializeArray< T >()

```
void SerializeArray< T > (
    ref T[] array,
    ArrayElementSerializer< T > serializer )
```

Serializes or deserializes an array of elements using a specified serializer.

Template Parameters

<i>T</i>	The type of the array elements.
----------	---------------------------------

Parameters

<i>array</i>	The array to serialize/deserialize.
<i>serializer</i>	The serializer to use for each element.

6.278.3.70 SerializeArrayLength< T >()

```
void SerializeArrayLength< T > (
    ref T[] array )
```

Serializes or deserializes the length of an array.

Template Parameters

<i>T</i>	The type of the array elements.
----------	---------------------------------

Parameters

<i>array</i>	The array whose length to serialize/deserialize.
--------------	--

6.278.3.71 SerializeBuffer() [1/8]

```
unsafe void SerializeBuffer (
    byte * buffer,
    int length )
```

Serializes or deserializes a buffer of byte values.

Parameters

<i>buffer</i>	The buffer of byte values to serialize/deserialize.
<i>length</i>	The length of the buffer.

6.278.3.72 SerializeBuffer() [2/8]

```
unsafe void SerializeBuffer (
    int * buffer,
    int length )
```

Serializes or deserializes a buffer of 32-bit integer values.

Parameters

<i>buffer</i>	The buffer of 32-bit integer values to serialize/deserialize.
<i>length</i>	The length of the buffer.

6.278.3.73 SerializeBuffer() [3/8]

```
unsafe void SerializeBuffer (
    long * buffer,
    int length )
```

Serializes or deserializes a buffer of 64-bit integer values.

Parameters

<i>buffer</i>	The buffer of 64-bit integer values to serialize/deserialize.
<i>length</i>	The length of the buffer.

6.278.3.74 SerializeBuffer() [4/8]

```
unsafe void SerializeBuffer (
    sbyte * buffer,
    int length )
```

Serializes or deserializes a buffer of signed byte values.

Parameters

<i>buffer</i>	The buffer of signed byte values to serialize/deserialize.
<i>length</i>	The length of the buffer.

6.278.3.75 SerializeBuffer() [5/8]

```
unsafe void SerializeBuffer (  
    short * buffer,  
    int length )
```

Serializes or deserializes a buffer of short values.

Parameters

<i>buffer</i>	The buffer of short values to serialize/deserialize.
<i>length</i>	The length of the buffer.

6.278.3.76 SerializeBuffer() [6/8]

```
unsafe void SerializeBuffer (  
    uint * buffer,  
    int length )
```

Serializes or deserializes a buffer of unsigned 32-bit integer values.

Parameters

<i>buffer</i>	The buffer of unsigned 32-bit integer values to serialize/deserialize.
<i>length</i>	The length of the buffer.

6.278.3.77 SerializeBuffer() [7/8]

```
unsafe void SerializeBuffer (  
    ulong * buffer,  
    int length )
```

Serializes or deserializes a buffer of unsigned 64-bit integer values.

Parameters

<i>buffer</i>	The buffer of unsigned 64-bit integer values to serialize/deserialize.
<i>length</i>	The length of the buffer.

6.278.3.78 SerializeBuffer() [8/8]

```
unsafe void SerializeBuffer (  
    
```

```

    ushort * buffer,
    int length )

```

Serializes or deserializes a buffer of unsigned short values.

Parameters

<i>buffer</i>	The buffer of unsigned short values to serialize/deserialize.
<i>length</i>	The length of the buffer.

6.278.3.79 SetBuffer() [1/2]

```

void SetBuffer (
    byte[] arr )

```

Sets the internal buffer to the specified byte array.

Parameters

<i>arr</i>	The byte array to use as the internal buffer.
------------	---

6.278.3.80 SetBuffer() [2/2]

```

void SetBuffer (
    byte[] arr,
    int size )

```

Sets the internal buffer to the specified byte array and size.

Parameters

<i>arr</i>	The byte array to use as the internal buffer.
<i>size</i>	The size of the byte array in bytes.

6.278.3.81 ToArray()

```

byte [] ToArray ( )

```

Converts the internal buffer to a byte array.

Returns

A byte array containing the data in the internal buffer.

6.278.3.82 WriteBool()

```
bool WriteBool (
    bool value )
```

Writes a boolean value to the stream.

Parameters

<i>value</i>	The boolean value to write.
--------------	-----------------------------

Returns

The boolean value that was written.

6.278.3.83 WriteBoolean()

```
bool WriteBoolean (
    bool value )
```

Writes a boolean value to the stream.

Parameters

<i>value</i>	The boolean value to write.
--------------	-----------------------------

Returns

The boolean value that was written.

6.278.3.84 WriteByte() [1/2]

```
void WriteByte (
    byte value )
```

Writes a byte value to the stream.

Parameters

<i>value</i>	The byte value to write.
--------------	--------------------------

6.278.3.85 WriteByte() [2/2]

```
void WriteByte (
    byte value,
    int bits )
```

Writes a byte value to the stream with a specified number of bits.

Parameters

<i>value</i>	The byte value to write.
<i>bits</i>	The number of bits to write.

6.278.3.86 WriteByteArray() [1/3]

```
void WriteByteArray (
    byte[] from )
```

Writes a byte array to the stream.

Parameters

<i>from</i>	The byte array to write.
-------------	--------------------------

6.278.3.87 WriteByteArray() [2/3]

```
void WriteByteArray (
    byte[] from,
    int count )
```

Writes a specified number of bytes from a byte array to the stream.

Parameters

<i>from</i>	The byte array to write from.
<i>count</i>	The number of bytes to write.

6.278.3.88 WriteByteArray() [3/3]

```
void WriteByteArray (
    byte[] from,
```



```
int offset,
int count )
```

Writes a specified number of bytes from a byte array to the stream starting at a given offset.

Parameters

<i>from</i>	The byte array to write from.
<i>offset</i>	The starting offset in the byte array.
<i>count</i>	The number of bytes to write.

6.278.3.89 WriteByteArrayLengthPrefixed() [1/2]

```
void WriteByteArrayLengthPrefixed (
    byte[] array )
```

Writes a byte array to the stream with a length prefix.

Parameters

<i>array</i>	The byte array to write.
--------------	--------------------------

6.278.3.90 WriteByteArrayLengthPrefixed() [2/2]

```
void WriteByteArrayLengthPrefixed (
    byte[] array,
    int maxLength )
```

Writes a byte array to the stream with a length prefix and a specified maximum length.

Parameters

<i>array</i>	The byte array to write.
<i>maxLength</i>	The maximum length of the byte array to write.

6.278.3.91 WriteByteAt()

```
static void WriteByteAt (
    byte[] data,
    int ptr,
    int bits,
    byte value ) [static]
```

Writes a byte value at a specified bit position in a byte array.

Parameters

<i>data</i>	The byte array to write to.
<i>ptr</i>	The bit position to start writing at.
<i>bits</i>	The number of bits to write.
<i>value</i>	The byte value to write.

6.278.3.92 WriteChar()

```
void WriteChar (  
    Char value )
```

Writes a character value to the stream.

Parameters

<i>value</i>	The character value to write.
--------------	-------------------------------

6.278.3.93 WriteDouble()

```
void WriteDouble (  
    double value )
```

Writes a 64-bit floating point value to the stream.

Parameters

<i>value</i>	The 64-bit floating point value to write.
--------------	---

6.278.3.94 WriteFloat()

```
void WriteFloat (  
    float value )
```

Writes a 32-bit floating point value to the stream.

Parameters

<i>value</i>	The 32-bit floating point value to write.
--------------	---

6.278.3.95 WriteGuid()

```
void WriteGuid (
    Guid guid )
```

Writes a GUID to the stream.

Parameters

<i>guid</i>	The GUID to write.
-------------	--------------------

6.278.3.96 WriteInt() [1/2]

```
void WriteInt (
    int value )
```

Writes a 32-bit integer value to the stream.

Parameters

<i>value</i>	The 32-bit integer value to write.
--------------	------------------------------------

6.278.3.97 WriteInt() [2/2]

```
void WriteInt (
    int value,
    int bits )
```

Writes a 32-bit integer value to the stream with a specified number of bits.

Parameters

<i>value</i>	The 32-bit integer value to write.
<i>bits</i>	The number of bits to write.

6.278.3.98 WriteInt_Shifted()

```
void WriteInt_Shifted (
    int value,
    int bits,
    int shift )
```

Writes a shifted 32-bit integer value to the stream with a specified number of bits.

Parameters

<i>value</i>	The 32-bit integer value to write.
<i>bits</i>	The number of bits to write.
<i>shift</i>	The number of bits to shift.

6.278.3.99 WriteLong() [1/2]

```
void WriteLong (  
    long value )
```

Writes a signed 64-bit integer value to the stream.

Parameters

<i>value</i>	The signed 64-bit integer value to write.
--------------	---

6.278.3.100 WriteLong() [2/2]

```
void WriteLong (  
    long value,  
    int bits )
```

Writes a signed 64-bit integer value to the stream with a specified number of bits.

Parameters

<i>value</i>	The signed 64-bit integer value to write.
<i>bits</i>	The number of bits to write.

6.278.3.101 WriteSByte()

```
void WriteSByte (  
    sbyte value )
```

Writes a signed byte value to the stream.

Parameters

<i>value</i>	The signed byte value to write.
--------------	---------------------------------

6.278.3.102 WriteShort() [1/2]

```
void WriteShort (
    short value )
```

Writes a short value to the stream.

Parameters

<i>value</i>	The short value to write.
--------------	---------------------------

6.278.3.103 WriteShort() [2/2]

```
void WriteShort (
    short value,
    int bits )
```

Writes a short value to the stream with a specified number of bits.

Parameters

<i>value</i>	The short value to write.
<i>bits</i>	The number of bits to write.

6.278.3.104 WriteString() [1/2]

```
void WriteString (
    string value )
```

Writes a string to the stream using UTF-8 encoding.

Parameters

<i>value</i>	The string to write.
--------------	----------------------

6.278.3.105 WriteString() [2/2]

```
void WriteString (
    string value,
    Encoding encoding )
```

Writes a string to the stream using the specified encoding.

Parameters

<i>value</i>	The string to write.
<i>encoding</i>	The encoding to use for the string.

6.278.3.106 WriteUInt() [1/2]

```
void WriteUInt (
    uint value )
```

Writes an unsigned 32-bit integer value to the stream.

Parameters

<i>value</i>	The unsigned 32-bit integer value to write.
--------------	---

6.278.3.107 WriteUInt() [2/2]

```
void WriteUInt (
    uint value,
    int bits )
```

Writes an unsigned 32-bit integer value to the stream with a specified number of bits.

Parameters

<i>value</i>	The unsigned 32-bit integer value to write.
<i>bits</i>	The number of bits to write.

6.278.3.108 WriteULong() [1/2]

```
void WriteULong (
    ulong value )
```

Writes an unsigned 64-bit integer value to the stream.

Parameters

<i>value</i>	The unsigned 64-bit integer value to write.
--------------	---

6.278.3.109 WriteULong() [2/2]

```
void WriteULong (
    ulong value,
    int bits )
```

Writes an unsigned 64-bit integer value to the stream with a specified number of bits.

Parameters

<i>value</i>	The unsigned 64-bit integer value to write.
<i>bits</i>	The number of bits to write.

6.278.3.110 WriteUShort() [1/2]

```
void WriteUShort (
    ushort value )
```

Writes an unsigned short value to the stream.

Parameters

<i>value</i>	The unsigned short value to write.
--------------	------------------------------------

6.278.3.111 WriteUShort() [2/2]

```
void WriteUShort (
    ushort value,
    int bits )
```

Writes an unsigned short value to the stream with a specified number of bits.

Parameters

<i>value</i>	The unsigned short value to write.
<i>bits</i>	The number of bits to write.

6.278.4 Property Documentation

6.278.4.1 BytesRequired

```
int BytesRequired [get]
```

Size of written buffer in BYTES Ammount of bytes required considering the total of written bytes

6.278.4.2 Capacity

```
int Capacity [get]
```

Total Size in BYTES of the Buffer

6.278.4.3 Data

```
Byte [] Data [get]
```

[Internal](#) Byte Array

6.278.4.4 Done

```
bool Done [get]
```

Signal if the buffer was completely written

6.278.4.5 IsEvenBytes

```
bool IsEvenBytes [get]
```

Gets a value indicating whether the current position is at an even byte boundary.

6.278.4.6 Overflowing

```
bool Overflowing [get]
```

Signal if the buffer is overflowing

6.278.4.7 Position

int Position [get], [set]

Current read/write position in BITS inside the Buffer

6.278.4.8 Reading

bool Reading [get], [set]

Signal if the Buffer is in Read Mode

6.278.4.9 Size

int Size [get], [set]

Total size in BITS of the buffer

6.278.4.10 Writing

bool Writing [get], [set]

Signal if the Buffer is in Write Mode

6.279 ICommunicator Interface Reference

Interface for a Communicator

Public Member Functions

- bool [Poll](#) ()
Check if there are data package to be retrieved
- void [PushPackage](#) (int senderActor, int eventCode, object data)
Push a new Package into the communicator queues
- int [ReceivePackage](#) (out int senderActor, byte *buffer, int bufferLength)
Retrieve a Data Package
- void [RegisterPackageCallback](#)< T > (Action< int, T > callback)
Register a callback for a specific Message Type
- void [SendMessage](#) (int targetActor, [IMessage](#) message)
Send a Protocol Message using the communicator system
- bool [SendPackage](#) (byte code, int targetActor, bool reliable, byte *buffer, int bufferLength)
Sends a package data using the communication system
- void [Service](#) ()
Step the Communicator internals

Properties

- int `CommunicatorID` [get]
Represents the current ID of the communicator.

6.279.1 Detailed Description

Interface for a Communicator

6.279.2 Member Function Documentation

6.279.2.1 Poll()

```
bool Poll ( )
```

Check if there are data package to be retrieved

Returns

True if the internal buffer has pending data

6.279.2.2 PushPackage()

```
void PushPackage (
    int senderActor,
    int eventCode,
    object data )
```

Push a new Package into the communicator queues

Parameters

<i>senderActor</i>	Data Sender Actor
<i>eventCode</i>	Event Code of the Package
<i>data</i>	Package

6.279.2.3 ReceivePackage()

```
int ReceivePackage (
    out int senderActor,
```

```

byte * buffer,
int bufferLength )

```

Retrieve a Data Package

Parameters

<i>senderActor</i>	Data Package Sender
<i>buffer</i>	Buffer to be filled with the Data
<i>bufferLength</i>	Buffer length

Returns

Total number of bytes written to buffer

6.279.2.4 RegisterPackageCallback< T >()

```

void RegisterPackageCallback< T > (
    Action< int, T > callback )

```

Register a callback for a specific Message Type

Template Parameters

<i>T</i>	Message Type
----------	--------------

Parameters

<i>callback</i>	Callback to be invoked when a Message of type T is received
-----------------	---

Type Constraints

***T* : IMessage**

6.279.2.5 SendMessage()

```

void SendMessage (
    int targetActor,
    IMessage message )

```

Send a [Protocol](#) Message using the communicator system

Parameters

<i>targetActor</i>	Target Actor of the Protocol Message
<i>message</i>	Protocol Message to be sent

6.279.2.6 SendPackage()

```
bool SendPackage (
    byte code,
    int targetActor,
    bool reliable,
    byte * buffer,
    int bufferLength )
```

Sends a package data using the communication system

Parameters

<i>code</i>	Event Code used to send the Package
<i>targetActor</i>	Target Actor of the Package
<i>reliable</i>	Flag if this Package should be sent reliably
<i>buffer</i>	Data Buffer
<i>bufferLength</i>	Buffer Length

6.279.2.7 Service()

```
void Service ( )
```

Step the Communicator internals

6.279.3 Property Documentation**6.279.3.1 CommunicatorID**

```
int CommunicatorID [get]
```

Represents the current ID of the communicator.

6.280 IMessage Interface Reference

Represents a [Protocol](#) Message

6.280.1 Detailed Description

Represents a [Protocol](#) Message

Used to tag the Messages in [ICommunicator](#).

6.281 Ptr Struct Reference

[Ptr](#)

Inherits [IEquatable](#)< [Ptr](#) >, and [INetworkStruct](#).

Classes

- class [EqualityComparer](#)
Ptr Equality Comparer

Public Member Functions

- override bool [Equals](#) (object obj)
Check Ptr equality
- bool [Equals](#) ([Ptr](#) other)
Check Ptr equality
- override int [GetHashCode](#) ()
Ptr Hash Code, same as Address
- override string [ToString](#) ()
Ptr to String

Static Public Member Functions

- static implicit [operator bool](#) ([Ptr](#) a)
Implicit Bool Operator Check if Address is not 0
- static bool [operator!=](#) ([Ptr](#) a, [Ptr](#) b)
Implicit Ptr Not Equals Operator
- static [Ptr operator+](#) ([Ptr](#) p, int v)
Implicit Ptr Sum Operator
- static [Ptr operator-](#) ([Ptr](#) p, int v)
Implicit Ptr Subtraction Operator
- static bool [operator==](#) ([Ptr](#) a, [Ptr](#) b)
Implicit Ptr Equals Operator

Public Attributes

- int [Address](#)
Ptr Address

Static Public Attributes

- const int [SIZE](#) = 4
Ptr Size

Properties

- static [Ptr Null](#) [get]
Null Ptr

6.281.1 Detailed Description

[Ptr](#)

6.281.2 Member Function Documentation

6.281.2.1 Equals() [1/2]

```
override bool Equals (  
    object obj )
```

Check [Ptr](#) equality

Parameters

<i>obj</i>	Any object reference
------------	----------------------

Returns

True if obj is a [Ptr](#) and points to the same Address

6.281.2.2 Equals() [2/2]

```
bool Equals (  
    Ptr other )
```

Check [Ptr](#) equality

Parameters

<i>other</i>	Ptr Ref
--------------	-------------------------

Returns

True if points to the same Address

6.281.2.3 GetHashCode()

```
override int GetHashCode ( )
```

[Ptr](#) Hash Code, same as [Address](#)

Returns

[Address](#)

6.281.2.4 operator bool()

```
static implicit operator bool (
    Ptr a ) [static]
```

Implicit Bool Operator Check if [Address](#) is not 0

Parameters

<i>a</i>	Ptr to check
----------	------------------------------

Returns

True if [Address](#) is not 0

6.281.2.5 operator"!=(())

```
static bool operator!= (
    Ptr a,
    Ptr b ) [static]
```

Implicit [Ptr](#) Not Equals Operator

Parameters

<i>a</i>	Ptr A
<i>b</i>	Ptr B

Returns

True if [Address](#) is not the same

6.281.2.6 operator+()

```
static Ptr operator+ (  
    Ptr p,  
    int v ) [static]
```

Implicit [Ptr](#) Sum Operator

Parameters

<i>p</i>	Ptr to add to
<i>v</i>	Value to add

Returns

[Ptr](#) with [Address](#) increased by *v*

6.281.2.7 operator-()

```
static Ptr operator- (  
    Ptr p,  
    int v ) [static]
```

Implicit [Ptr](#) Subtraction Operator

Parameters

<i>p</i>	Ptr to subtract from
<i>v</i>	Value to subtract

Returns

[Ptr](#) with [Address](#) decreased by *v*

6.281.2.8 operator==()

```
static bool operator== (  
    Ptr a,  
    Ptr b ) [static]
```

Implicit [Ptr](#) Equals Operator

Parameters

<i>a</i>	Ptr A
<i>b</i>	Ptr B

Returns

True if [Address](#) is the same

6.281.2.9 ToString()

```
override string ToString ( )
```

[Ptr](#) to String

Returns

[Address](#) in Hexadecimal format

6.281.3 Member Data Documentation**6.281.3.1 Address**

```
int Address
```

[Ptr](#) Address

6.281.3.2 SIZE

```
const int SIZE = 4 [static]
```

[Ptr](#) Size

6.281.4 Property Documentation

6.281.4.1 Null

`Ptr Null` [static], [get]

Null `Ptr`

6.282 Ptr.EqualityComparer Class Reference

`Ptr Equality Comparer`

Inherits `IEqualityComparer< Ptr >`.

Public Member Functions

- bool `Equals` (`Ptr x`, `Ptr y`)
Ptr Equality Comparer
- int `GetHashCode` (`Ptr obj`)
Get Hash Code

6.282.1 Detailed Description

`Ptr Equality Comparer`

6.282.2 Member Function Documentation

6.282.2.1 Equals()

```
bool Equals (  
    Ptr x,  
    Ptr y )
```

`Ptr Equality Comparer`

Parameters

<code>x</code>	<code>Ptr X</code>
<code>y</code>	<code>Ptr Y</code>

Returns

True if point to the same Address

6.282.2.2 GetHashCode()

```
int GetHashCode (
    Ptr obj )
```

Get Hash Code

Parameters

<i>obj</i>	Ptr
------------	-----

Returns

Ptr Address

6.283 QuaternionCompressed Struct Reference

Represents a compressed Quaternion value for network transmission.

Inherits [INetworkStruct](#), and [IEquatable< QuaternionCompressed >](#).

Public Member Functions

- override bool [Equals](#) (object obj)
Checks if the provided object is a [QuaternionCompressed](#) instance and if it's equal to the current [QuaternionCompressed](#) instance.
- bool [Equals](#) ([QuaternionCompressed](#) other)
Checks if the current [QuaternionCompressed](#) instance is equal to the other [QuaternionCompressed](#) instance.
- override int [GetHashCode](#) ()
Returns the hash code for the current [QuaternionCompressed](#) instance.

Static Public Member Functions

- static implicit [operator Quaternion](#) ([QuaternionCompressed](#) q)
Implicit conversion from [QuaternionCompressed](#) to Quaternion.
- static implicit [operator QuaternionCompressed](#) (Quaternion v)
Implicit conversion from Quaternion to [QuaternionCompressed](#).
- static bool [operator!=](#) ([QuaternionCompressed](#) left, [QuaternionCompressed](#) right)
Inequality operator for [QuaternionCompressed](#) struct.
- static bool [operator==](#) ([QuaternionCompressed](#) left, [QuaternionCompressed](#) right)
Equality operator for [QuaternionCompressed](#) struct.

Public Attributes

- int [wEncoded](#)
Encoded value of the w component.
- int [xEncoded](#)
Encoded value of the x component.
- int [yEncoded](#)
Encoded value of the y component.
- int [zEncoded](#)
Encoded value of the z component.

Properties

- float [W](#) [get, set]
Gets or sets the w component.
- float [X](#) [get, set]
Gets or sets the x component.
- float [Y](#) [get, set]
Gets or sets the y component.
- float [Z](#) [get, set]
Gets or sets the z component.

6.283.1 Detailed Description

Represents a compressed Quaternion value for network transmission.

6.283.2 Member Function Documentation

6.283.2.1 Equals() [1/2]

```
override bool Equals (  
    object obj )
```

Checks if the provided object is a [QuaternionCompressed](#) instance and if it's equal to the current [QuaternionCompressed](#) instance.

Parameters

<i>obj</i>	The object to compare with the current QuaternionCompressed instance.
------------	---

Returns

True if the provided object is a [QuaternionCompressed](#) instance and it's equal to the current [QuaternionCompressed](#) instance, otherwise false.

6.283.2.2 Equals() [2/2]

```
bool Equals (  
    QuaternionCompressed other )
```

Checks if the current [QuaternionCompressed](#) instance is equal to the other [QuaternionCompressed](#) instance.

Parameters

<i>other</i>	The other QuaternionCompressed instance to compare with the current QuaternionCompressed instance.
--------------	--

Returns

True if the values of both [QuaternionCompressed](#) instances are equal, otherwise false.

6.283.2.3 GetHashCode()

```
override int GetHashCode ( )
```

Returns the hash code for the current [QuaternionCompressed](#) instance.

Returns

A hash code for the current [QuaternionCompressed](#) instance.

6.283.2.4 operator Quaternion()

```
static implicit operator Quaternion (
    QuaternionCompressed q ) [static]
```

Implicit conversion from [QuaternionCompressed](#) to Quaternion.

Parameters

<i>q</i>	The QuaternionCompressed instance to convert.
----------	---

Returns

The decompressed Quaternion value of the [QuaternionCompressed](#) instance.

6.283.2.5 operator QuaternionCompressed()

```
static implicit operator QuaternionCompressed (
    Quaternion v ) [static]
```

Implicit conversion from Quaternion to [QuaternionCompressed](#).

Parameters

<i>v</i>	The Quaternion value to convert.
----------	----------------------------------

Returns

A new [QuaternionCompressed](#) instance with the compressed value of the Quaternion.

6.283.2.6 operator"!=()

```
static bool operator!= (
    QuaternionCompressed left,
    QuaternionCompressed right ) [static]
```

Inequality operator for [QuaternionCompressed](#) struct.

Parameters

<i>left</i>	First QuaternionCompressed instance.
<i>right</i>	Second QuaternionCompressed instance.

Returns

True if the value of the first [QuaternionCompressed](#) instance is not equal to the value of the second [QuaternionCompressed](#) instance, otherwise false.

6.283.2.7 operator=="()

```
static bool operator== (
    QuaternionCompressed left,
    QuaternionCompressed right ) [static]
```

Equality operator for [QuaternionCompressed](#) struct.

Parameters

<i>left</i>	First QuaternionCompressed instance.
<i>right</i>	Second QuaternionCompressed instance.

Returns

True if the value of the first [QuaternionCompressed](#) instance is equal to the value of the second [QuaternionCompressed](#) instance, otherwise false.

6.283.3 Member Data Documentation

6.283.3.1 wEncoded

```
int wEncoded
```

Encoded value of the w component.

6.283.3.2 xEncoded

```
int xEncoded
```

Encoded value of the x component.

6.283.3.3 yEncoded

```
int yEncoded
```

Encoded value of the y component.

6.283.3.4 zEncoded

```
int zEncoded
```

Encoded value of the z component.

6.283.4 Property Documentation

6.283.4.1 W

```
float W [get], [set]
```

Gets or sets the w component.

6.283.4.2 X

float X [get], [set]

Gets or sets the x component.

6.283.4.3 Y

float Y [get], [set]

Gets or sets the y component.

6.283.4.4 Z

float Z [get], [set]

Gets or sets the z component.

6.284 RangeExAttribute Class Reference

Represents an attribute that specifies a range of values for a field or property.

Inherits [DrawerPropertyAttribute](#).

Public Member Functions

- [RangeExAttribute](#) (double min, double max)
Initializes a new instance of the [RangeExAttribute](#) class with the specified minimum and maximum values.

Public Attributes

- bool [ClampMax](#) = true
Gets or sets a value indicating whether the maximum value should be clamped.
- bool [ClampMin](#) = true
Gets or sets a value indicating whether the minimum value should be clamped.
- bool [UseSlider](#) = true
Gets or sets a value indicating whether a slider should be used for the range.

Properties

- double [Max](#) [get]
Gets the maximum value of the range.
- double [Min](#) [get]
Gets the minimum value of the range.

6.284.1 Detailed Description

Represents an attribute that specifies a range of values for a field or property.

6.284.2 Constructor & Destructor Documentation

6.284.2.1 RangeExAttribute()

```
RangeExAttribute (
    double min,
    double max )
```

Initializes a new instance of the [RangeExAttribute](#) class with the specified minimum and maximum values.

Parameters

<i>min</i>	The minimum value of the range.
<i>max</i>	The maximum value of the range.

6.284.3 Member Data Documentation

6.284.3.1 ClampMax

```
bool ClampMax = true
```

Gets or sets a value indicating whether the maximum value should be clamped.

6.284.3.2 ClampMin

```
bool ClampMin = true
```

Gets or sets a value indicating whether the minimum value should be clamped.

6.284.3.3 UseSlider

```
bool UseSlider = true
```

Gets or sets a value indicating whether a slider should be used for the range.

6.284.4 Property Documentation

6.284.4.1 Max

double Max [get]

Gets the maximum value of the range.

6.284.4.2 Min

double Min [get]

Gets the minimum value of the range.

6.285 ReadOnlyAttribute Class Reference

Attribute used to mark a field as read-only.

Inherits [DecoratingPropertyAttribute](#).

Properties

- bool [InEditMode](#) = true [get, set]
Should the field be read-only in edit mode?
- bool [InPlayMode](#) = true [get, set]
Should the field be read-only in play mode?

Additional Inherited Members

6.285.1 Detailed Description

Attribute used to mark a field as read-only.

6.285.2 Property Documentation

6.285.2.1 InEditMode

```
bool InEditMode = true [get], [set]
```

Should the field be read-only in edit mode?

6.285.2.2 InPlayMode

```
bool InPlayMode = true [get], [set]
```

Should the field be read-only in play mode?

6.286 ReadWriteUtils Class Reference

Provides utility methods for reading and writing data.

Static Public Member Functions

- static float [ReadFloat](#) (int *data)
Reads a float value from the provided memory location.
- static Quaternion [ReadQuaternion](#) (int *data)
Reads a Quaternion value from the provided memory location.
- static Vector2 [ReadVector2](#) (int *data)
Reads a Vector2 value from the provided memory location.
- static Vector3 [ReadVector3](#) (int *data)
Reads a Vector3 value from the provided memory location.
- static Vector4 [ReadVector4](#) (int *data)
Reads a Vector4 value from the provided memory location.
- static void [WriteFloat](#) (int *data, float f)
Writes a float value to the provided memory location.
- static void [WriteQuaternion](#) (int *data, Quaternion value)
Writes a Quaternion value to the provided memory location.
- static void [WriteVector2](#) (int *data, Vector2 value)
Writes a Vector2 value to the provided memory location.
- static void [WriteVector3](#) (int *data, Vector3 value)
Writes a Vector3 value to the provided memory location.
- static void [WriteVector4](#) (int *data, Vector4 value)
Writes a Vector4 value to the provided memory location.

Static Public Attributes

- const float [ACCURACY](#) = 1 << 10
Accuracy of floating point values when serialized.

6.286.1 Detailed Description

Provides utility methods for reading and writing data.

6.286.2 Member Function Documentation

6.286.2.1 ReadFloat()

```
static float ReadFloat (  
    int * data ) [static]
```

Reads a float value from the provided memory location.

Parameters

<i>data</i>	The memory location to read from.
-------------	-----------------------------------

Returns

The float value read from the memory location.

6.286.2.2 ReadQuaternion()

```
static Quaternion ReadQuaternion (  
    int * data ) [static]
```

Reads a Quaternion value from the provided memory location.

Parameters

<i>data</i>	The memory location to read from.
-------------	-----------------------------------

Returns

The Quaternion value read from the memory location.

6.286.2.3 ReadVector2()

```
static Vector2 ReadVector2 (  
    int * data ) [static]
```

Reads a Vector2 value from the provided memory location.

Parameters

<i>data</i>	The memory location to read from.
-------------	-----------------------------------

Returns

The Vector2 value read from the memory location.

6.286.2.4 ReadVector3()

```
static Vector3 ReadVector3 (  
    int * data ) [static]
```

Reads a Vector3 value from the provided memory location.

Parameters

<i>data</i>	The memory location to read from.
-------------	-----------------------------------

Returns

The Vector3 value read from the memory location.

6.286.2.5 ReadVector4()

```
static Vector4 ReadVector4 (  
    int * data ) [static]
```

Reads a Vector4 value from the provided memory location.

Parameters

<i>data</i>	The memory location to read from.
-------------	-----------------------------------

Returns

The Vector4 value read from the memory location.

6.286.2.6 WriteFloat()

```
static void WriteFloat (  
    int * data,  
    float f ) [static]
```

Writes a float value to the provided memory location.

Parameters

<i>data</i>	The memory location to write to.
<i>f</i>	The float value to write.

6.286.2.7 WriteQuaternion()

```
static void WriteQuaternion (
    int * data,
    Quaternion value ) [static]
```

Writes a Quaternion value to the provided memory location.

Parameters

<i>data</i>	The memory location to write to.
<i>value</i>	The Quaternion value to write.

6.286.2.8 WriteVector2()

```
static void WriteVector2 (
    int * data,
    Vector2 value ) [static]
```

Writes a Vector2 value to the provided memory location.

Parameters

<i>data</i>	The memory location to write to.
<i>value</i>	The Vector2 value to write.

6.286.2.9 WriteVector3()

```
static void WriteVector3 (
    int * data,
    Vector3 value ) [static]
```

Writes a Vector3 value to the provided memory location.

Parameters

<i>data</i>	The memory location to write to.
<i>value</i>	The Vector3 value to write.

6.286.2.10 WriteVector4()

```
static void WriteVector4 (
    int * data,
    Vector4 value ) [static]
```

Writes a Vector4 value to the provided memory location.

Parameters

<i>data</i>	The memory location to write to.
<i>value</i>	The Vector4 value to write.

6.286.3 Member Data Documentation**6.286.3.1 ACCURACY**

```
const float ACCURACY = 1 << 10 [static]
```

Accuracy of floating point values when serialized.

6.287 ReadWriteUtilsForWeaver Class Reference

Provides utility methods for reading and writing data.

Static Public Member Functions

- static unsafe int [GetByteArrayHashCode](#) (byte *ptr, int length)
- static int [GetByteCountUtf8NoHash](#) (string value)
 - Gets the byte count of a string in UTF8 format without a hash.*
- static int [GetStringHashCode](#) (string value, int maxLength)
- static int [GetWordCountString](#) (int capacity, bool withCaching)
 - Gets the word count of a string with optional caching.*
- static bool [ReadBoolean](#) (int *data)
 - Reads a boolean value from the provided memory location.*
- static unsafe int [ReadStringUtf32NoHash](#) (int *ptr, int maxLength, out string result)

- Reads a string from the provided memory location in UTF32 format without a hash.*

 - static unsafe int [ReadStringUtf32WithHash](#) (int *ptr, int maxLength, ref string cache)
- Reads a string from the provided memory location in UTF32 format with a hash.*

 - static int [ReadStringUtf8NoHash](#) (void *source, out string result)
- Reads a string from the provided memory location in UTF8 format without a hash.*

 - static int [VerifyRawNetworkUnwrap< T >](#) (int actual, int maxBytes)
- Verifies the byte count of a network unwrapped object. Throws an exception if the actual byte count exceeds the maximum allowed byte count.*

 - static int [VerifyRawNetworkWrap< T >](#) (int actual, int maxBytes)
- Verifies the byte count of a network wrapped object. Throws an exception if the actual byte count exceeds the maximum allowed byte count.*

 - static void [WriteBoolean](#) (int *data, bool value)
- Writes a boolean value to the provided memory location.*

 - static unsafe int [WriteStringUtf32NoHash](#) (int *ptr, int maxLength, string value)
- Writes a string to the provided memory location in UTF32 format without a hash.*

 - static unsafe int [WriteStringUtf32WithHash](#) (int *ptr, int maxLength, string value, ref string cache)
- Writes a string to the provided memory location in UTF32 format with a hash.*

 - static int [WriteStringUtf8NoHash](#) (void *destination, string str)
- Writes a string to the provided memory location in UTF8 format without a hash.*

6.287.1 Detailed Description

Provides utility methods for reading and writing data.

6.287.2 Member Function Documentation

6.287.2.1 GetByteArrayHashCode()

```
static unsafe int GetByteArrayHashCode (
    byte * ptr,
    int length ) [static]
```

Parameters

<i>ptr</i>	
<i>length</i>	

Returns

6.287.2.2 GetByteCountUtf8NoHash()

```
static int GetByteCountUtf8NoHash (
    string value ) [static]
```

Gets the byte count of a string in UTF8 format without a hash.

Parameters

<i>value</i>	The string to get the byte count of.
--------------	--------------------------------------

Returns

The byte count of the string in UTF8 format.

6.287.2.3 GetStringHashCode()

```
static int GetStringHashCode (
    string value,
    int maxLength ) [static]
```

Parameters

<i>value</i>	
<i>maxLength</i>	

Returns

6.287.2.4 GetWordCountString()

```
static int GetWordCountString (
    int capacity,
    bool withCaching ) [static]
```

Gets the word count of a string with optional caching.

Parameters

<i>capacity</i>	The capacity of the string.
<i>withCaching</i>	Indicates whether caching is used.

Returns

The word count of the string.

6.287.2.5 ReadBoolean()

```
static bool ReadBoolean (
    int * data ) [static]
```

Reads a boolean value from the provided memory location.

Parameters

<i>data</i>	The memory location to read from.
-------------	-----------------------------------

Returns

The boolean value read from the memory location.

6.287.2.6 ReadStringUtf32NoHash()

```
static unsafe int ReadStringUtf32NoHash (
    int * ptr,
    int maxLength,
    out string result ) [static]
```

Reads a string from the provided memory location in UTF32 format without a hash.

Parameters

<i>ptr</i>	The memory location to read from.
<i>maxLength</i>	The maximum length of the string.
<i>result</i>	The string read from the memory location.

Returns

The number of bytes read.

6.287.2.7 ReadStringUtf32WithHash()

```
static unsafe int ReadStringUtf32WithHash (
    int * ptr,
```

```
int maxLength,  
ref string cache ) [static]
```

Reads a string from the provided memory location in UTF32 format with a hash.

Parameters

<i>ptr</i>	The memory location to read from.
<i>maxLength</i>	The maximum length of the string.
<i>cache</i>	A reference to a cache string. This will be updated with the read string if it matches the cached hashcode.

Returns

The number of bytes read.

6.287.2.8 ReadStringUtf8NoHash()

```
static int ReadStringUtf8NoHash (
    void * source,
    out string result ) [static]
```

Reads a string from the provided memory location in UTF8 format without a hash.

Parameters

<i>source</i>	The memory location to read from.
<i>result</i>	The string read from the memory location.

Returns

The number of bytes read.

6.287.2.9 VerifyRawNetworkUnwrap< T >()

```
static int VerifyRawNetworkUnwrap< T > (
    int actual,
    int maxBytes ) [static]
```

Verifies the byte count of a network unwrapped object. Throws an exception if the actual byte count exceeds the maximum allowed byte count.

Template Parameters

<i>T</i>	The type of the network unwrapped object.
----------	---

Parameters

<i>actual</i>	The actual byte count.
---------------	------------------------

Parameters

<i>maxBytes</i>	The maximum allowed byte count.
-----------------	---------------------------------

Returns

The actual byte count if it does not exceed the maximum allowed byte count.

6.287.2.10 VerifyRawNetworkWrap< T >()

```
static int VerifyRawNetworkWrap< T > (
    int actual,
    int maxBytes ) [static]
```

Verifies the byte count of a network wrapped object. Throws an exception if the actual byte count exceeds the maximum allowed byte count.

Template Parameters

<i>T</i>	The type of the network wrapped object.
----------	---

Parameters

<i>actual</i>	The actual byte count.
<i>maxBytes</i>	The maximum allowed byte count.

Returns

The actual byte count if it does not exceed the maximum allowed byte count.

6.287.2.11 WriteBoolean()

```
static void WriteBoolean (
    int * data,
    bool value ) [static]
```

Writes a boolean value to the provided memory location.

Parameters

<i>data</i>	The memory location to write to.
<i>value</i>	The boolean value to write.

6.287.2.12 WriteStringUtf32NoHash()

```
static unsafe int WriteStringUtf32NoHash (  
    int * ptr,  
    int maxLength,  
    string value ) [static]
```

Writes a string to the provided memory location in UTF32 format without a hash.

Parameters

<i>ptr</i>	The memory location to write to.
<i>maxLength</i>	The maximum length of the string.
<i>value</i>	The string to write.

Returns

The number of bytes written.

6.287.2.13 WriteStringUtf32WithHash()

```
static unsafe int WriteStringUtf32WithHash (  
    int * ptr,  
    int maxLength,  
    string value,  
    ref string cache ) [static]
```

Writes a string to the provided memory location in UTF32 format with a hash.

Parameters

<i>ptr</i>	The memory location to write to.
<i>maxLength</i>	The maximum length of the string.
<i>value</i>	The string to write.
<i>cache</i>	A reference to a cache string. This will be updated with the trimmed value of the input string.

Returns

The number of bytes written.

6.287.2.14 WriteStringUtf8NoHash()

```
static int WriteStringUtf8NoHash (  

```

```
void * destination,
string str ) [static]
```

Writes a string to the provided memory location in UTF8 format without a hash.

Parameters

<i>destination</i>	The memory location to write to.
<i>str</i>	The string to write.

Returns

The number of bytes written.

6.288 ReflectionUtils Class Reference

Provides utility methods for reflection.

Static Public Member Functions

- static IEnumerable< Type > [GetAllNetworkBehaviourTypes](#) ()
Gets all types that are assignable from [NetworkBehaviour](#) from all assemblies.
- static IEnumerable< Type > [GetAllSimulationBehaviourTypes](#) ()
Gets all types that are assignable from [SimulationBehaviour](#) from all assemblies.
- static IEnumerable< Assembly > [GetAllWeavedAssemblies](#) ()
Gets all assemblies that have been weaved.
- static IEnumerable< Type > [GetAllWeavedNetworkBehaviourTypes](#) ()
Gets all types that are assignable from [NetworkBehaviour](#) from all weaved assemblies.
- static IEnumerable< Type > [GetAllWeavedSimulationBehaviourTypes](#) ()
Gets all types that are assignable from [SimulationBehaviour](#) from all weaved assemblies.
- static IEnumerable< Type > [GetAllWeaverGeneratedTypes](#) ()
Gets all types that have the [WeaverGeneratedAttribute](#) from all weaved assemblies.
- static T [GetCustomAttributeOrThrow](#)< T > (this MemberInfo member, bool inherit)
Retrieves a custom attribute of type T from the provided member.
- static [NetworkBehaviourWeavedAttribute](#) [GetWeavedAttributeOrThrow](#) (Type type)
Gets the [NetworkBehaviourWeavedAttribute](#) for the specified type. Throws an [InvalidOperationException](#) if the type has not been weaved.

6.288.1 Detailed Description

Provides utility methods for reflection.

6.288.2 Member Function Documentation

6.288.2.1 GetAllNetworkBehaviourTypes()

```
static IEnumerable<Type> GetAllNetworkBehaviourTypes ( ) [static]
```

Gets all types that are assignable from [NetworkBehaviour](#) from all assemblies.

Returns

An IEnumerable of all types that are assignable from [NetworkBehaviour](#).

6.288.2.2 GetAllSimulationBehaviourTypes()

```
static IEnumerable<Type> GetAllSimulationBehaviourTypes ( ) [static]
```

Gets all types that are assignable from [SimulationBehaviour](#) from all assemblies.

Returns

An IEnumerable of all types that are assignable from [SimulationBehaviour](#).

6.288.2.3 GetAllWeavedAssemblies()

```
static IEnumerable<Assembly> GetAllWeavedAssemblies ( ) [static]
```

Gets all assemblies that have been weaved.

Returns

An IEnumerable of all weaved assemblies.

6.288.2.4 GetAllWeavedNetworkBehaviourTypes()

```
static IEnumerable<Type> GetAllWeavedNetworkBehaviourTypes ( ) [static]
```

Gets all types that are assignable from [NetworkBehaviour](#) from all weaved assemblies.

Returns

An IEnumerable of all types that are assignable from [NetworkBehaviour](#) in weaved assemblies.

6.288.2.5 GetAllWeavedSimulationBehaviourTypes()

```
static IEnumerable<Type> GetAllWeavedSimulationBehaviourTypes ( ) [static]
```

Gets all types that are assignable from [SimulationBehaviour](#) from all weaved assemblies.

Returns

An IEnumerable of all types that are assignable from [SimulationBehaviour](#) in weaved assemblies.

6.288.2.6 GetAllWeaverGeneratedTypes()

```
static IEnumerable<Type> GetAllWeaverGeneratedTypes ( ) [static]
```

Gets all types that have the [WeaverGeneratedAttribute](#) from all weaved assemblies.

Returns

An IEnumerable of all types that have the [WeaverGeneratedAttribute](#) in weaved assemblies.

6.288.2.7 GetCustomAttributeOrThrow< T >()

```
static T GetCustomAttributeOrThrow< T > (
    this MemberInfo member,
    bool inherit ) [static]
```

Retrieves a custom attribute of type T from the provided member.

Template Parameters

<i>T</i>	The type of the attribute to retrieve. Must be a subclass of Attribute.
----------	---

Parameters

<i>member</i>	The member to retrieve the attribute from.
<i>inherit</i>	Specifies whether to search this member's inheritance chain to find the attributes.

Returns

The custom attribute of type T.

Exceptions

<i>ArgumentOutOfRangeException</i>	Thrown when the provided member does not have an attribute of type T.
<i>InvalidOperationException</i>	Thrown when the provided member has more than one attribute of type T.

Type Constraints

T : *Attribute***6.288.2.8** GetWeavedAttributeOrThrow()

```
static NetworkBehaviourWeavedAttribute GetWeavedAttributeOrThrow (
    Type type ) [static]
```

Gets the [NetworkBehaviourWeavedAttribute](#) for the specified type. Throws an `InvalidOperationException` if the type has not been weaved.

Parameters

<i>type</i>	The type to get the NetworkBehaviourWeavedAttribute for.
-------------	--

Returns

The [NetworkBehaviourWeavedAttribute](#) for the specified type.

Exceptions

<i>InvalidOperationException</i>	Thrown when the type has not been weaved.
----------------------------------	---

6.289 RenderAttribute Class Reference

Override default render settings for [Networked] properties.

Inherits `Attribute`.

Public Member Functions

- [RenderAttribute](#) ()
Default constructor for [RenderAttribute](#)
- [RenderAttribute](#) ([RenderTimeframe](#) timeframe, [RenderSource](#) source)
[RenderAttribute](#) Constructor

Properties

- string [Method](#) [get, set]
- [RenderSource](#) [Source](#) [get, set]
- [RenderTimeframe](#) [Timeframe](#) [get, set]

6.289.1 Detailed Description

Override default render settings for [Networked] properties.

6.289.2 Constructor & Destructor Documentation

6.289.2.1 RenderAttribute() [1/2]

```
RenderAttribute ( )
```

Default constructor for [RenderAttribute](#)

6.289.2.2 RenderAttribute() [2/2]

```
RenderAttribute (
    RenderTimeframe timeframe,
    RenderSource source )
```

[RenderAttribute](#) Constructor

Parameters

<i>timeframe</i>	RenderTimeframe reference
<i>source</i>	RenderSource reference

6.289.3 Property Documentation

6.289.3.1 Method

```
string Method [get], [set]
```

Override the default interpolation method for this property. The method's signature must match:

```
static T MethodName(T from, T to, float alpha) { /* ... */ }
```

6.289.3.2 Source

```
RenderSource Source [get], [set]
```

Force this property to be rendered using this [RenderSource](#) (in the chosen [RenderTimeframe](#)).

This setting is prioritized over [NetworkBehaviour](#) and [NetworkObject](#) overrides.

6.289.3.3 Timeframe

`RenderTimeframe` Timeframe [get], [set]

Force this property to be rendered in this [RenderTimeframe](#).

This setting is prioritized over [NetworkBehaviour](#) and [NetworkObject](#) overrides.

6.290 RenderTimeline Struct Reference

Can be used to acquire interpolated data for different points in time.

Static Public Member Functions

- static void [GetRenderBuffers](#) ([NetworkBehaviour](#) behaviour, out [NetworkBehaviourBuffer](#) from, out [NetworkBehaviourBuffer](#) to, out float alpha)

Get the render data for the given [NetworkBehaviour](#).

6.290.1 Detailed Description

Can be used to acquire interpolated data for different points in time.

6.290.2 Member Function Documentation

6.290.2.1 GetRenderBuffers()

```
static void GetRenderBuffers (
    NetworkBehaviour behaviour,
    out NetworkBehaviourBuffer from,
    out NetworkBehaviourBuffer to,
    out float alpha ) [static]
```

Get the render data for the given [NetworkBehaviour](#).

Parameters

<i>behaviour</i>	Network behaviour to get render data for.
<i>from</i>	Render data for the previous point in time.
<i>to</i>	Render data for the next point in time.
<i>alpha</i>	Interpolation alpha.

6.291 RenderWeavedAttribute Class Reference

Render Weaved Attribute

Inherits [Attribute](#).

Public Member Functions

- [RenderWeavedAttribute](#) ()
RenderWeavedAttribute Constructor

6.291.1 Detailed Description

Render Weaved Attribute

6.291.2 Constructor & Destructor Documentation

6.291.2.1 RenderWeavedAttribute()

[RenderWeavedAttribute](#) ()

[RenderWeavedAttribute](#) Constructor

6.292 ResolveNetworkPrefabSourceAttribute Class Reference

Resolve Network Prefab Source Attribute

Inherits [PropertyAttribute](#).

6.292.1 Detailed Description

Resolve Network Prefab Source Attribute

6.293 RpcAttribute Class Reference

Flags a method as being a networked Remote Procedure Call. Only usable in a [NetworkBehaviour](#). Calls to this method (from the indicated allowed [RpcSources](#)) will generate a network message, which will execute the method remotely on the indicated [RpcTargets](#). The RPC method can include an empty [RpcInfo](#) argument, that will include meta information about the RPC on the receiving peer.

Inherits [Attribute](#).

Public Member Functions

- [RpcAttribute](#) ()
Constructor for RpcAttributes.
- [RpcAttribute](#) ([RpcSources](#) sources, [RpcTargets](#) targets)
Constructor for RpcAttributes.

Static Public Attributes

- const int [MaxPayloadSize](#) = [SimulationMessage.MAX_PAYLOAD_SIZE](#)
Maximum allowed size for the payload of the RPC message.

Properties

- [RpcChannel](#) [Channel](#) = [RpcChannel.Reliable](#) [get, set]
Specifies which RpcChannel to use. Default value is [RpcChannel.Reliable](#)
- [RpcHostMode](#) [HostMode](#) = [RpcHostMode.SourceIsServer](#) [get, set]
Options for when the game is run in [SimulationModes.Host](#) mode and RPC is invoked by the host.
- bool [InvokeLocal](#) = true [get, set]
Indicates if the method should be called locally (on the RPC caller). This happens immediately. Default value is true.
- int [Sources](#) [get]
The legal [RpcSources](#) types that can trigger this Rpc. Cast to int. Default value is (int)[RpcSources.All](#).
- int [Targets](#) [get]
The [RpcTargets](#) types that will receive and invoke this method. Cast to int. Default value is (int)[RpcTargets.All](#).
- bool [TickAligned](#) = true [get, set]
Indicates if this RPC's execution will be postponed until the local simulation catches up with the sender's [Tick](#) number. Even if set to false, the order of Rpcs is always preserved. Rpcs are deferred until all preceding Rpcs have executed. Default value is true.

6.293.1 Detailed Description

Flags a method as being a networked Remote Procedure Call. Only usable in a [NetworkBehaviour](#). Calls to this method (from the indicated allowed [RpcSources](#)) will generate a network message, which will execute the method remotely on the indicated [RpcTargets](#). The RPC method can include an empty [RpcInfo](#) argument, that will include meta information about the RPC on the receiving peer.

Example:

```
| [Rpc(RpcSources.All, RpcTargets.All, InvokeLocal = false, InvokeResim =
false, Channel = RpcChannel.Reliable, TickAligned = true)]
| public void RPC_Configure(NetworkObject no, string name, Color color, RpcInfo
info = default) { } To target a specific Player, use the RpcTargetAttribute: | [Rpc] | public
void RpcFoo([RpcTarget] PlayerRef targetPlayer) {} Use RpcInvokeInfo as a return value
to access meta information about the RPC send attempt, such as failure to send reasons, message size, etc.
```

Non-static RPCs are only valid on a [NetworkBehaviour](#). Static RPCs can be implemented on [SimulationBehaviours](#), and do not require a [NetworkObject](#) instance. Static RPC require the first argument to be [NetworkRunner](#).

```
Static RPC Example: | [Rpc] | public static void RPC_Configure(NetworkRunner
runner) { }
```

6.293.2 Constructor & Destructor Documentation

6.293.2.1 `RpcAttribute()` [1/2]

```
RpcAttribute ( )
```

Constructor for `RpcAttributes`.

6.293.2.2 `RpcAttribute()` [2/2]

```
RpcAttribute (
    RpcSources sources,
    RpcTargets targets )
```

Constructor for `RpcAttributes`.

Parameters

<code>sources</code>	The legal <code>RpcSources</code> types that can trigger this <code>Rpc</code> . Default is <code>RpcSources.All</code>
<code>targets</code>	The <code>RpcTargets</code> types that will receive and invoke this method. Default is <code>RpcTargets.All</code>

6.293.3 Member Data Documentation

6.293.3.1 `MaxPayloadSize`

```
const int MaxPayloadSize = SimulationMessage.MAX_PAYLOAD_SIZE [static]
```

Maximum allowed size for the payload of the RPC message.

6.293.4 Property Documentation

6.293.4.1 `Channel`

```
RpcChannel Channel = RpcChannel.Reliable [get], [set]
```

Specifies which `RpcChannel` to use. Default value is `RpcChannel.Reliable`

6.293.4.2 HostMode

```
RpcHostMode HostMode = RpcHostMode.SourceIsServer [get], [set]
```

Options for when the game is run in [SimulationModes.Host](#) mode and RPC is invoked by the host.

6.293.4.3 InvokeLocal

```
bool InvokeLocal = true [get], [set]
```

Indicates if the method should be called locally (on the RPC caller). This happens immediately. Default value is true.

6.293.4.4 Sources

```
int Sources [get]
```

The legal [RpcSources](#) types that can trigger this Rpc. Cast to int. Default value is (int)[RpcSources.All](#).

6.293.4.5 Targets

```
int Targets [get]
```

The [RpcTargets](#) types that will receive and invoke this method. Cast to int. Default value is (int)[RpcTargets.All](#).

6.293.4.6 TickAligned

```
bool TickAligned = true [get], [set]
```

Indicates if this RPC's execution will be postponed until the local simulation catches up with the sender's [Tick](#) number. Even if set to false, the order of Rpcs is always preserved. Rpcs are deferred until all preceding Rpcs have executed. Default value is true.

6.294 RpcHeader Struct Reference

Header for RPC messages.

Public Member Functions

- override string [ToString](#) ()
Returns a string that represents the current [RpcHeader](#).

Static Public Member Functions

- static [RpcHeader Create](#) (int staticRpcKey)
Creates a new [RpcHeader](#) with the provided staticRpcKey.
- static [RpcHeader Create](#) ([NetworkId](#) id, int behaviour, int method)
Creates a new [RpcHeader](#) with the provided [NetworkId](#), behaviour, and method.
- static [RpcHeader Read](#) (byte *data, out int size)
Reads the [RpcHeader](#) from the provided byte pointer.
- static int [ReadSize](#) (byte *data)
Reads the size of the [RpcHeader](#) from the provided byte pointer.
- static int [Write](#) ([RpcHeader](#) header, byte *data)
Writes the [RpcHeader](#) to the provided byte pointer.

Public Attributes

- ushort [Behaviour](#)
The behaviour associated with the RPC message.
- ushort [Method](#)
The method associated with the RPC message.
- [NetworkId Object](#)
The [NetworkId](#) of the object associated with the RPC message.

Static Public Attributes

- const int [SIZE](#) = [NetworkId](#).SIZE + 2 + 2
The size of the [RpcHeader](#) structure in bytes.

6.294.1 Detailed Description

Header for RPC messages.

6.294.2 Member Function Documentation

6.294.2.1 Create() [1/2]

```
static RpcHeader Create (  
    int staticRpcKey ) [static]
```

Creates a new [RpcHeader](#) with the provided staticRpcKey.

Parameters

<i>staticRpcKey</i>	The staticRpcKey associated with the RPC message.
---------------------	---

Returns

Returns a new [RpcHeader](#) with the provided staticRpcKey.

6.294.2.2 Create() [2/2]

```
static RpcHeader Create (  
    NetworkId id,  
    int behaviour,  
    int method ) [static]
```

Creates a new [RpcHeader](#) with the provided [NetworkId](#), behaviour, and method.

Parameters

<i>id</i>	The NetworkId of the object associated with the RPC message.
<i>behaviour</i>	The behaviour associated with the RPC message.
<i>method</i>	The method associated with the RPC message.

Returns

Returns a new [RpcHeader](#) with the provided parameters.

6.294.2.3 Read()

```
static RpcHeader Read (  
    byte * data,  
    out int size ) [static]
```

Reads the [RpcHeader](#) from the provided byte pointer.

Parameters

<i>data</i>	The byte pointer to read the RpcHeader from.
<i>size</i>	The size of the RpcHeader structure in bytes.

Returns

Returns the [RpcHeader](#) read from the byte pointer.

6.294.2.4 ReadSize()

```
static int ReadSize (  
    byte * data ) [static]
```

Reads the size of the [RpcHeader](#) from the provided byte pointer.

Parameters

<i>data</i>	The byte pointer to read the RpcHeader size from.
-------------	---

Returns

Returns the size of the [RpcHeader](#) structure in bytes.

6.294.2.5 ToString()

```
override string ToString ( )
```

Returns a string that represents the current [RpcHeader](#).

Returns

Returns a string that represents the current [RpcHeader](#).

6.294.2.6 Write()

```
static int Write (  
    RpcHeader header,  
    byte * data ) [static]
```

Writes the [RpcHeader](#) to the provided byte pointer.

Parameters

<i>header</i>	The RpcHeader to write.
<i>data</i>	The byte pointer to write the RpcHeader to.

Returns

Returns the size of the [RpcHeader](#) structure in bytes.

6.294.3 Member Data Documentation

6.294.3.1 Behaviour

ushort [Behaviour](#)

The behaviour associated with the RPC message.

6.294.3.2 Method

ushort [Method](#)

The method associated with the RPC message.

6.294.3.3 Object

[NetworkId](#) [Object](#)

The [NetworkId](#) of the object associated with the RPC message.

6.294.3.4 SIZE

```
const int SIZE = NetworkId.SIZE + 2 + 2 [static]
```

The size of the [RpcHeader](#) structure in bytes.

6.295 RpcInfo Struct Reference

[RpcInfo](#) is a struct that contains information about the RPC message.

Public Member Functions

- override string [ToString](#) ()
Returns a string that represents the current [RpcInfo](#).

Static Public Member Functions

- static [RpcInfo FromLocal](#) ([NetworkRunner](#) runner, [RpcChannel](#) channel, [RpcHostMode](#) hostMode)
Creates a new [RpcInfo](#) instance for a local RPC message.
- static unsafe [RpcInfo FromMessage](#) ([NetworkRunner](#) runner, [SimulationMessage](#) *message, [RpcHostMode](#) hostMode)
Creates a new [RpcInfo](#) instance from a [SimulationMessage](#).

Public Attributes

- [RpcChannel Channel](#)
Represents the channel through which the RPC message was sent.
- bool [IsInvokeLocal](#)
Indicates whether the RPC message is invoked locally.
- [PlayerRef Source](#)
Represents the player who sent the RPC message.
- [Tick Tick](#)
Represents the tick at which the RPC message was sent.

6.295.1 Detailed Description

[RpcInfo](#) is a struct that contains information about the RPC message.

6.295.2 Member Function Documentation

6.295.2.1 FromLocal()

```
static RpcInfo FromLocal (
    NetworkRunner runner,
    RpcChannel channel,
    RpcHostMode hostMode ) [static]
```

Creates a new [RpcInfo](#) instance for a local RPC message.

Parameters

<i>runner</i>	The NetworkRunner associated with the RPC message.
<i>channel</i>	The RpcChannel through which the RPC message was sent.
<i>hostMode</i>	The RpcHostMode of the RPC message.

Returns

Returns a new [RpcInfo](#) instance with the provided parameters.

6.295.2.2 FromMessage()

```
static unsafe RpcInfo FromMessage (
    NetworkRunner runner,
    SimulationMessage * message,
    RpcHostMode hostMode ) [static]
```

Creates a new [RpcInfo](#) instance from a [SimulationMessage](#).

Parameters

<i>runner</i>	The NetworkRunner associated with the RPC message.
<i>message</i>	The SimulationMessage from which to create the RpcInfo instance.
<i>hostMode</i>	The RpcHostMode of the RPC message.

Returns

Returns a new [RpcInfo](#) instance with the provided parameters.

6.295.2.3 ToString()

```
override string ToString ( )
```

Returns a string that represents the current [RpcInfo](#).

Returns

Returns a string that represents the current [RpcInfo](#).

6.295.3 Member Data Documentation**6.295.3.1 Channel**

[RpcChannel](#) Channel

Represents the channel through which the RPC message was sent.

6.295.3.2 IsInvokeLocal

```
bool IsInvokeLocal
```

Indicates whether the RPC message is invoked locally.

6.295.3.3 Source

[PlayerRef](#) Source

Represents the player who sent the RPC message.

6.295.3.4 Tick

`Tick Tick`

Represents the tick at which the RPC message was sent.

6.296 RpcInvokeData Struct Reference

Represents the data required to invoke an RPC message.

Public Member Functions

- override string [ToString](#) ()
Returns a string that represents the current [RpcInvokeData](#).

Public Attributes

- [RpcInvokeDelegate Delegate](#)
Represents the delegate to be invoked for the RPC message.
- int [Key](#)
Represents the key associated with the RPC message.
- int [Sources](#)
Represents the sources of the RPC message.
- int [Targets](#)
Represents the targets of the RPC message.

6.296.1 Detailed Description

Represents the data required to invoke an RPC message.

6.296.2 Member Function Documentation

6.296.2.1 ToString()

```
override string ToString ( )
```

Returns a string that represents the current [RpcInvokeData](#).

Returns

Returns a string that represents the current [RpcInvokeData](#).

6.296.3 Member Data Documentation

6.296.3.1 Delegate

[RpcInvokeDelegate](#) Delegate

Represents the delegate to be invoked for the RPC message.

6.296.3.2 Key

int Key

Represents the key associated with the RPC message.

6.296.3.3 Sources

int Sources

Represents the sources of the RPC message.

6.296.3.4 Targets

int Targets

Represents the targets of the RPC message.

6.297 RpcInvokeInfo Struct Reference

May be used as an optional [RpcAttribute](#) return value. Contains meta data about the RPC send, such as failure to send reasons, culling, message size, etc.

Public Member Functions

- override string [ToString](#) ()
Returns a string that represents the current [RpcInvokeInfo](#).

Public Attributes

- [RpcLocalInvokeResult](#) [LocalInvokeResult](#)
Represents the result of the local RPC invocation.
- [RpcSendCullResult](#) [SendCullResult](#)
Represents the result of the RPC message send operation.
- [RpcSendResult](#) [SendResult](#)
Contains detailed information about the RPC send operation result.

6.297.1 Detailed Description

May be used as an optional [RpcAttribute](#) return value. Contains meta data about the RPC send, such as failure to send reasons, culling, message size, etc.

Example:

```
| [Rpc] | public RpcInvokeInfo RpcFoo(int value) { | return default; | } | |  
public override void FixedUpdateNetwork() { | var info = RpcFoo(); | Debug.  
Log(info); | }
```

6.297.2 Member Function Documentation

6.297.2.1 ToString()

```
override string ToString ( )
```

Returns a string that represents the current [RpcInvokeInfo](#).

6.297.3 Member Data Documentation

6.297.3.1 LocalInvokeResult

```
RpcLocalInvokeResult LocalInvokeResult
```

Represents the result of the local RPC invocation.

6.297.3.2 SendCullResult

```
RpcSendCullResult SendCullResult
```

Represents the result of the RPC message send operation.

6.297.3.3 SendResult

`RpcSendResult` SendResult

Contains detailed information about the RPC send operation result.

6.298 RpcSendResult Struct Reference

RPC send operation result information.

Public Member Functions

- override string `ToString` ()
Returns a string that represents the current `RpcSendResult`.

Public Attributes

- int `MessageSize`
The size of the RPC message.
- `RpcSendMessageResult` `Result`
Result flags for the RPC send operation.

6.298.1 Detailed Description

RPC send operation result information.

6.298.2 Member Function Documentation

6.298.2.1 ToString()

```
override string ToString ( )
```

Returns a string that represents the current `RpcSendResult`.

6.298.3 Member Data Documentation

6.298.3.1 MessageSize

int MessageSize

The size of the RPC message.

6.298.3.2 Result

[RpcSendMessageResult](#) Result

Result flags for the RPC send operation.

6.299 RpcTargetAttribute Class Reference

RPC attribute used to indicate a specific target player for an RPC when sending from one player to another. RPC is sent to the server, and then is forwarded to the specified player. Usage:

Inherits Attribute.

Public Member Functions

- [RpcTargetAttribute](#) ()
RPC Attribute constructor.

6.299.1 Detailed Description

RPC attribute used to indicate a specific target player for an RPC when sending from one player to another. RPC is sent to the server, and then is forwarded to the specified player. Usage:

```
| [Rpc] | public void RpcFoo([RpcTarget] PlayerRef targetPlayer) { }
```

6.299.2 Constructor & Destructor Documentation

6.299.2.1 RpcTargetAttribute()

[RpcTargetAttribute](#) ()

RPC Attribute constructor.

6.300 RuntimeUnityFlagsSetup Class Reference

Contains methods to setup runtime flags for Unity.

Static Public Member Functions

- static void [Check_ENABLE_IL2CPP](#) ()
Checks if the ENABLE_IL2CPP flag is set and updates the flagsDotNetVersion accordingly.
- static void [Check_ENABLE_MONO](#) ()
Checks if the ENABLE_MONO flag is set and updates the flagsDotNetVersion accordingly.
- static void [Check_NET_4_6](#) ()
Checks if the NET_4_6 flag is set and updates the flagsDotNetVersion accordingly.
- static void [Check_NET_STANDARD_2_0](#) ()
Checks if the NET_STANDARD_2_0 flag is set and updates the flagsDotNetVersion accordingly.
- static void [Check_NETFX_CORE](#) ()
Checks if the NETFX_CORE flag is set and updates the flagsDotNetVersion accordingly.
- static void [Check_UNITY_2019_4_OR_NEWER](#) ()
Checks if the UNITY_2019_4_OR_NEWER flag is set and updates the flagsDotNetVersion accordingly.
- static void [Check_UNITY_EDITOR](#) ()
Checks if the UNITY_EDITOR flag is set and updates the flagsDotNetVersion accordingly.
- static void [Check_UNITY_GAMECORE](#) ()
Checks if the UNITY_GAMECORE flag is set and updates the flagsDotNetVersion accordingly.
- static void [Check_UNITY_SWITCH](#) ()
Checks if the UNITY_SWITCH flag is set and updates the flagsDotNetVersion accordingly.
- static void [Check_UNITY_WEBGL](#) ()
Checks if the UNITY_WEBGL flag is set and updates the flagsDotNetVersion accordingly.
- static void [Check_UNITY_XBOXONE](#) ()
Checks if the UNITY_XBOXONE flag is set and updates the flagsDotNetVersion accordingly.
- static void [Reset](#) ()
Resets all runtime flags to their default values.

6.300.1 Detailed Description

Contains methods to setup runtime flags for Unity.

6.300.2 Member Function Documentation

6.300.2.1 Check_ENABLE_IL2CPP()

```
static void Check_ENABLE_IL2CPP ( ) [static]
```

Checks if the ENABLE_IL2CPP flag is set and updates the flagsDotNetVersion accordingly.

6.300.2.2 Check_ENABLE_MONO()

```
static void Check_ENABLE_MONO ( ) [static]
```

Checks if the ENABLE_MONO flag is set and updates the flagsDotNetVersion accordingly.

6.300.2.3 Check_NET_4_6()

```
static void Check_NET_4_6 ( ) [static]
```

Checks if the NET_4_6 flag is set and updates the flagsDotNetVersion accordingly.

6.300.2.4 Check_NET_STANDARD_2_0()

```
static void Check_NET_STANDARD_2_0 ( ) [static]
```

Checks if the NET_STANDARD_2_0 flag is set and updates the flagsDotNetVersion accordingly.

6.300.2.5 Check_NETFX_CORE()

```
static void Check_NETFX_CORE ( ) [static]
```

Checks if the NETFX_CORE flag is set and updates the flagsDotNetVersion accordingly.

6.300.2.6 Check_UNITY_2019_4_OR_NEWER()

```
static void Check_UNITY_2019_4_OR_NEWER ( ) [static]
```

Checks if the UNITY_2019_4_OR_NEWER flag is set and updates the flagsDotNetVersion accordingly.

6.300.2.7 Check_UNITY_EDITOR()

```
static void Check_UNITY_EDITOR ( ) [static]
```

Checks if the UNITY_EDITOR flag is set and updates the flagsDotNetVersion accordingly.

6.300.2.8 Check_UNITY_GAMECORE()

```
static void Check_UNITY_GAMECORE ( ) [static]
```

Checks if the UNITY_GAMECORE flag is set and updates the flagsDotNetVersion accordingly.

6.300.2.9 Check_UNITY_SWITCH()

```
static void Check_UNITY_SWITCH ( ) [static]
```

Checks if the UNITY_SWITCH flag is set and updates the flagsDotNetVersion accordingly.

6.300.2.10 Check_UNITY_WEBGL()

```
static void Check_UNITY_WEBGL ( ) [static]
```

Checks if the UNITY_WEBGL flag is set and updates the flagsDotNetVersion accordingly.

6.300.2.11 Check_UNITY_XBOXONE()

```
static void Check_UNITY_XBOXONE ( ) [static]
```

Checks if the UNITY_XBOXONE flag is set and updates the flagsDotNetVersion accordingly.

6.300.2.12 Reset()

```
static void Reset ( ) [static]
```

Resets all runtime flags to their default values.

6.301 SceneLoadDoneArgs Struct Reference

Struct that contains information about a scene after it has been loaded.

Public Member Functions

- [SceneLoadDoneArgs](#) ([SceneRef](#) sceneRef, [NetworkObject](#)[] sceneObjects, [Scene](#) scene=default, [GameObject](#)[] rootGameObjects=default)

Constructs a new [SceneLoadDoneArgs](#) struct.

Public Attributes

- readonly `GameObject[]` [RootGameObjects](#)
Array of root GameObjects present in the loaded Unity scene.
- readonly `Scene` [Scene](#)
The loaded Unity scene.
- readonly `NetworkObject[]` [SceneObjects](#)
Array of NetworkObjects present in the loaded scene.
- readonly [SceneRef](#) `SceneRef`
Reference to the loaded scene.

6.301.1 Detailed Description

Struct that contains information about a scene after it has been loaded.

6.301.2 Constructor & Destructor Documentation

6.301.2.1 SceneLoadDoneArgs()

```
SceneLoadDoneArgs (
    SceneRef sceneRef,
    NetworkObject[] sceneObjects,
    Scene scene = default,
    GameObject[] rootGameObjects = default )
```

Constructs a new [SceneLoadDoneArgs](#) struct.

Parameters

<i>sceneRef</i>	Reference to the loaded scene.
<i>sceneObjects</i>	Array of NetworkObjects present in the loaded scene.
<i>scene</i>	The loaded Unity scene.
<i>rootGameObjects</i>	Array of root GameObjects present in the loaded Unity scene.

6.301.3 Member Data Documentation

6.301.3.1 RootGameObjects

```
readonly GameObject [] RootGameObjects
```

Array of root GameObjects present in the loaded Unity scene.

6.301.3.2 Scene

```
readonly Scene Scene
```

The loaded Unity scene.

6.301.3.3 SceneObjects

```
readonly NetworkObject [] SceneObjects
```

Array of NetworkObjects present in the loaded scene.

6.301.3.4 SceneRef

```
readonly SceneRef SceneRef
```

Reference to the loaded scene.

6.302 ScenePathAttribute Class Reference

Specifies that a string field represents a scene path.

Inherits [DrawerPropertyAttribute](#).

6.302.1 Detailed Description

Specifies that a string field represents a scene path.

6.303 SceneRef Struct Reference

Scene reference struct. Can be used to reference a scene by index or by path.

Inherits [INetworkStruct](#), and [IEquatable< SceneRef >](#).

Public Member Functions

- override bool [Equals](#) (object obj)
Determines whether the specified object is equal to the current [SceneRef](#).
- bool [Equals](#) ([SceneRef](#) other)
Determines whether the specified [SceneRef](#) is equal to the current [SceneRef](#).
- override int [GetHashCode](#) ()
Serves as the default hash function.
- bool [IsPath](#) (string path)
Checks if the [SceneRef](#) corresponds to a specific path.
- override string [ToString](#) ()
Returns a string that represents the current [SceneRef](#).
- string [ToString](#) (bool brackets, bool prefix)
Returns a string that represents the current [SceneRef](#), with optional formatting.

Static Public Member Functions

- static [SceneRef FromIndex](#) (int index)
Creates a [SceneRef](#) from an index.
- static [SceneRef FromPath](#) (string path)
Creates a scene ref from a path. The most common use case for this method is when using Unity's addressable scenes. The path is hashed (31 bit), so on rare occasion there may be a hash collision. In such case consider renaming a scene or construct your own hash and use [FromRaw](#). To check if a scene ref is was created for a specific path, use [IsPath](#).
- static [SceneRef FromRaw](#) (uint rawValue)
Creates a [SceneRef](#) from a raw value.
- static bool [operator!=](#) ([SceneRef](#) a, [SceneRef](#) b)
Returns true if the values are not equal.
- static bool [operator==](#) ([SceneRef](#) a, [SceneRef](#) b)
Returns true if the values are equal.
- static [SceneRef Parse](#) (string str)
Parses a [SceneRef](#) from a string.

Public Attributes

- uint [RawValue](#)
The raw value of the [SceneRef](#). This can represent either an index or a path hash, depending on the flag.

Static Public Attributes

- const uint [FLAG_ADDRESSABLE](#) = 1u << 31
A constant representing the flag for addressable scenes.
- const int [SIZE](#) = 4
The size of the [SceneRef](#) structure in bytes.

Properties

- int [AsIndex](#) [get]
Returns lower 32 bits as an index.
- uint [AsPathHash](#) [get]
Gets the path hash of the [SceneRef](#).
- bool [IsIndex](#) [get]
Returns true if this scene ref is an index.
- bool [IsValid](#) [get]
If this scene index is valid
- static [SceneRef None](#) [get]
None scene

6.303.1 Detailed Description

Scene reference struct. Can be used to reference a scene by index or by path.

6.303.2 Member Function Documentation

6.303.2.1 Equals() [1/2]

```
override bool Equals (  
    object obj )
```

Determines whether the specified object is equal to the current [SceneRef](#).

Parameters

<i>obj</i>	The object to compare with the current SceneRef .
------------	---

Returns

true if the specified object is equal to the current [SceneRef](#); otherwise, false.

6.303.2.2 Equals() [2/2]

```
bool Equals (  
    SceneRef other )
```

Determines whether the specified [SceneRef](#) is equal to the current [SceneRef](#).

Parameters

<i>other</i>	The SceneRef to compare with the current SceneRef .
--------------	---

Returns

true if the specified [SceneRef](#) is equal to the current [SceneRef](#); otherwise, false.

6.303.2.3 FromIndex()

```
static SceneRef FromIndex (  
    int index ) [static]
```

Creates a [SceneRef](#) from an index.

Parameters

<i>index</i>	The index to create the SceneRef from.
--------------	--

Returns

A [SceneRef](#) that represents the index.

Exceptions

<i>ArgumentOutOfRangeException</i>	Thrown when the index is less than 0 or equal to <code>int.MaxValue</code> .
------------------------------------	--

6.303.2.4 FromPath()

```
static SceneRef FromPath (  
    string path ) [static]
```

Creates a scene ref from a path. The most common use case for this method is when using Unity's addressable scenes. The path is hashed (31 bit), so on rare occasion there may be a hash collision. In such case consider renaming a scene or construct your own hash and use [FromRaw](#). To check if a scene ref is was created for a specific path, use [IsPath](#).

Parameters

<i>path</i>	The path to create the SceneRef from.
-------------	---

Returns

A [SceneRef](#) that represents the path.

6.303.2.5 FromRaw()

```
static SceneRef FromRaw (
    uint rawValue ) [static]
```

Creates a [SceneRef](#) from a raw value.

Parameters

<i>rawValue</i>	The raw value to create the SceneRef from.
-----------------	--

Returns

A [SceneRef](#) that represents the raw value.

6.303.2.6 GetHashCode()

```
override int GetHashCode ( )
```

Serves as the default hash function.

Returns

A hash code for the current [SceneRef](#).

6.303.2.7 IsPath()

```
bool IsPath (
    string path )
```

Checks if the [SceneRef](#) corresponds to a specific path.

Parameters

<i>path</i>	The path to check.
-------------	--------------------

Returns

true if the [SceneRef](#) corresponds to the path; otherwise, false.

6.303.2.8 operator"!=()

```
static bool operator!= (
    SceneRef a,
    SceneRef b ) [static]
```

Returns true if the values are not equal.

Parameters

<i>a</i>	SceneRef a
<i>b</i>	SceneRef b

Returns

true if the values are not equal; otherwise, false.

6.303.2.9 operator=="()

```
static bool operator== (
    SceneRef a,
    SceneRef b ) [static]
```

Returns true if the values are equal.

Parameters

<i>a</i>	SceneRef a
<i>b</i>	SceneRef b

Returns

true if the values are equal; otherwise, false.

6.303.2.10 Parse()

```
static SceneRef Parse (
    string str ) [static]
```

Parses a [SceneRef](#) from a string.

Parameters

<i>str</i>	
------------	--

Returns

6.303.2.11 ToString() [1/2]

```
override string ToString ( )
```

Returns a string that represents the current [SceneRef](#).

Returns

A string that represents the current [SceneRef](#).

6.303.2.12 ToString() [2/2]

```
string ToString (
    bool brackets,
    bool prefix )
```

Returns a string that represents the current [SceneRef](#), with optional formatting.

Parameters

<i>brackets</i>	If true, the string will be enclosed in brackets.
<i>prefix</i>	If true, the string will be prefixed with "Scene:".

Returns

A string that represents the current [SceneRef](#), formatted according to the provided parameters.

6.303.3 Member Data Documentation**6.303.3.1 FLAG_ADDRESSABLE**

```
const uint FLAG_ADDRESSABLE = 1u << 31 [static]
```

A constant representing the flag for addressable scenes.

6.303.3.2 RawValue

```
uint RawValue
```

The raw value of the [SceneRef](#). This can represent either an index or a path hash, depending on the flag.

6.303.3.3 SIZE

```
const int SIZE = 4 [static]
```

The size of the [SceneRef](#) structure in bytes.

6.303.4 Property Documentation

6.303.4.1 AsIndex

```
int AsIndex [get]
```

Returns lower 32 bits as an index.

6.303.4.2 AsPathHash

```
uint AsPathHash [get]
```

Gets the path hash of the [SceneRef](#).

Exceptions

<i>InvalidOperationException</i>	Thrown when the SceneRef is an index, not a path.
----------------------------------	---

6.303.4.3 IsIndex

```
bool IsIndex [get]
```

Returns true if this scene ref is an index.

6.303.4.4 IsValid

```
bool IsValid [get]
```

If this scene index is valid

6.303.4.5 None

```
SceneRef None [static], [get]
```

None scene

6.304 ScriptHelpAttribute Class Reference

Defines the appearance of the script header in the Unity inspector.

Inherits [PropertyAttribute](#).

Properties

- [ScriptHeaderBackColor BackColor](#) = [ScriptHeaderBackColor.Gray](#) [get, set]
Color of the inspector header for this component type. None indicates no header graphic should be used.
- bool [Hide](#) [get, set]
Hide the script header in the Unity inspector.
- [ScriptHeaderStyle Style](#) = [ScriptHeaderStyle.Photon](#) [get, set]
- string [Url](#) [get, set]

6.304.1 Detailed Description

Defines the appearance of the script header in the Unity inspector.

6.304.2 Property Documentation

6.304.2.1 BackColor

```
ScriptHeaderBackColor BackColor = ScriptHeaderBackColor.Gray [get], [set]
```

Color of the inspector header for this component type. None indicates no header graphic should be used.

6.304.2.2 Hide

```
bool Hide [get], [set]
```

Hide the script header in the Unity inspector.

6.304.2.3 Style

```
ScriptHeaderStyle Style = ScriptHeaderStyle.Photon [get], [set]
```

6.304.2.4 Url

```
string Url [get], [set]
```

6.305 SerializableDictionary< TKey, TValue > Class Template Reference

A serializable dictionary.

Inherits SerializableDictionary, IDictionary< TKey, TValue >, and ISerializationCallbackReceiver.

Public Member Functions

- void [Add](#) (TKey key, TValue value)
Adds the specified key and value to the [SerializableDictionary](#).
- virtual void [Clear](#) ()
Removes all keys and values from the [SerializableDictionary](#).
- bool [ContainsKey](#) (TKey key)
Determines whether the [SerializableDictionary](#) contains the specified key.
- Dictionary< TKey, TValue >.Enumerator [GetEnumerator](#) ()
Returns an enumerator that iterates through the [SerializableDictionary](#).
- bool [Remove](#) (TKey key)
Removes the value with the specified key from the [SerializableDictionary](#).
- void [Reset](#) ()
Resets the [SerializableDictionary](#), clearing its internal dictionary.
- void [Store](#) ()
Stores the [SerializableDictionary](#)'s data into an array for serialization. This includes handling duplicates and null keys.
- bool [TryGetValue](#) (TKey key, out TValue value)
Gets the value associated with the specified key.

Static Public Member Functions

- static [SerializableDictionary](#)< TKey, TValue > [Create](#)< TKey, TValue > ()
Creates a new serializable dictionary.
- static [SerializableDictionary](#)< TKey, TValue > [Wrap](#) (Dictionary< TKey, TValue > dictionary)
Wraps an existing Dictionary into a [SerializableDictionary](#).

Static Public Attributes

- const string [EntryKeyPropertyPath](#) = nameof(Entry.Key)
The property path for the key in the Entry structure.
- const string [ItemsPropertyPath](#) = nameof(_items)
The property path for the items in the [SerializableDictionary](#).

Properties

- int [Count](#) [get]
Gets the number of key/value pairs contained in the [SerializableDictionary](#).
- bool [IsReadOnly](#) [get]
Gets a value indicating whether the [SerializableDictionary](#) is read-only. This value is always false.
- Dictionary< TKey, TValue >.KeyCollection [Keys](#) [get]
Gets a collection containing the keys in the [SerializableDictionary](#).
- TValue [this\[TKey key\]](#) [get, set]
Gets or sets the value associated with the specified key.
- Dictionary< TKey, TValue >.ValueCollection [Values](#) [get]
Gets a collection containing the values in the [SerializableDictionary](#).

6.305.1 Detailed Description

A serializable dictionary.

Template Parameters

<i>TKey</i>	The type of the dictionary key.
<i>TValue</i>	The type of the dictionary value.

This class is not thread-safe.

6.305.2 Member Function Documentation

6.305.2.1 Add()

```
void Add (
    TKey key,
    TValue value )
```

Adds the specified key and value to the [SerializableDictionary](#).

Parameters

<i>key</i>	The key of the element to add.
<i>value</i>	The value of the element to add.

6.305.2.2 Clear()

```
virtual void Clear ( ) [virtual]
```

Removes all keys and values from the [SerializableDictionary](#).

6.305.2.3 ContainsKey()

```
bool ContainsKey (
    TKey key )
```

Determines whether the [SerializableDictionary](#) contains the specified key.

Parameters

<i>key</i>	The key to locate in the SerializableDictionary .
------------	---

Returns

true if the [SerializableDictionary](#) contains an element with the specified key; otherwise, false.

6.305.2.4 Create< TKey, TValue >()

```
static SerializableDictionary<TKey, TValue> Create< TKey, TValue > ( ) [static]
```

Creates a new serializable dictionary.

Template Parameters

<i>TKey</i>	The type of the dictionary key.
<i>TValue</i>	The type of the dictionary value.

Returns

A new serializable dictionary.

6.305.2.5 GetEnumerator()

```
Dictionary<TKey, TValue>.Enumerator GetEnumerator ( )
```

Returns an enumerator that iterates through the [SerializableDictionary](#).

Returns

A Dictionary{TKey,TValue}.Enumerator structure for the [SerializableDictionary](#).

6.305.2.6 Remove()

```
bool Remove (
    TKey key )
```

Removes the value with the specified key from the [SerializableDictionary](#).

Parameters

<i>key</i>	The key of the element to remove.
------------	-----------------------------------

Returns

true if the element is successfully found and removed; otherwise, false. This method returns false if key is not found in the [SerializableDictionary](#).

6.305.2.7 Reset()

```
void Reset ( )
```

Resets the [SerializableDictionary](#), clearing its internal dictionary.

6.305.2.8 Store()

```
void Store ( )
```

Stores the [SerializableDictionary](#)'s data into an array for serialization. This includes handling duplicates and null keys.

6.305.2.9 TryGetValue()

```
bool TryGetValue (
    TKey key,
    out TValue value )
```

Gets the value associated with the specified key.

Parameters

<i>key</i>	The key of the value to get.
<i>value</i>	When this method returns, contains the value associated with the specified key, if the key is found; otherwise, the default value for the type of the value parameter. This parameter is passed uninitialized.

Returns

true if the [SerializableDictionary](#) contains an element with the specified key; otherwise, false.

6.305.2.10 Wrap()

```
static SerializableDictionary<TKey, TValue> Wrap (  
    Dictionary< TKey, TValue > dictionary ) [static]
```

Wraps an existing Dictionary into a [SerializableDictionary](#).

Parameters

<i>dictionary</i>	The Dictionary to be wrapped.
-------------------	-------------------------------

Returns

A new [SerializableDictionary](#) that wraps the provided Dictionary.

6.305.3 Member Data Documentation**6.305.3.1 EntryKeyPropertyPath**

```
const string EntryKeyPropertyPath = nameof(Entry.Key) [static]
```

The property path for the key in the Entry structure.

6.305.3.2 ItemsPropertyPath

```
const string ItemsPropertyPath = nameof(_items) [static]
```

The property path for the items in the [SerializableDictionary](#).

6.305.4 Property Documentation

6.305.4.1 Count

```
int Count [get]
```

Gets the number of key/value pairs contained in the [SerializableDictionary](#).

6.305.4.2 IsReadOnly

```
bool IsReadOnly [get]
```

Gets a value indicating whether the [SerializableDictionary](#) is read-only. This value is always false.

6.305.4.3 Keys

```
Dictionary<TKey, TValue>.KeyCollection Keys [get]
```

Gets a collection containing the keys in the [SerializableDictionary](#).

6.305.4.4 this[TKey key]

```
TValue this[TKey key] [get], [set]
```

Gets or sets the value associated with the specified key.

Parameters

<i>key</i>	The key of the value to get or set.
------------	-------------------------------------

Returns

The value associated with the specified key. If the specified key is not found, a get operation throws a [KeyNotFoundException](#), and a set operation creates a new element with the specified key.

6.305.4.5 Values

Dictionary<TKey, TValue>.ValueCollection Values [get]

Gets a collection containing the values in the [SerializableDictionary](#).

6.306 SerializableType< BaseType > Struct Template Reference

A System.Type wrapper that can be serialized.

Inherits [IEquatable< SerializableType >](#), and [IEquatable< SerializableType< BaseType >>](#).

Public Member Functions

- [SerializableType AsShort \(\)](#)
Converts [AssemblyQualifiedName](#) and returns a short form, without version, culture etc.
- [SerializableType< BaseType > AsShort \(\)](#)
- override bool [Equals](#) (object obj)
Returns true if obj is [SerializableType](#) and the [AssemblyQualifiedName](#) is the same.
- override bool [Equals](#) (object obj)
- bool [Equals](#) ([SerializableType](#) other)
Returns true if the [AssemblyQualifiedName](#) is the same.
- bool [Equals](#) ([SerializableType< BaseType >](#) other)
- override int [GetHashCode](#) ()
Returns the hash code of the [AssemblyQualifiedName](#).
- override int [GetHashCode](#) ()
- [SerializableType](#) (string type)
Create a new instance and stores type as [AssemblyQualifiedName](#).
- [SerializableType](#) (Type type)
Create a new instance and stores full Type.AssemblyQualifiedName. To use shorter form, use [AsShort](#).
- [SerializableType](#) (Type type)

Static Public Member Functions

- static string [GetShortAssemblyQualifiedName](#) (Type type)
Converts the Type.AssemblyQualifiedName to a shorter form, without version, culture etc.
- static implicit operator [SerializableType](#) (Type type)
Implicitly convert a Type to a [SerializableType](#).
- static implicit operator [SerializableType< BaseType >](#) (Type type)
- static implicit operator Type ([SerializableType](#) serializableType)
Implicitly convert a [SerializableType](#) to a Type.
- static implicit operator Type ([SerializableType< BaseType >](#) serializableType)

Public Attributes

- string [AssemblyQualifiedName](#)
Type's assembly qualified name.

Static Public Attributes

- static readonly Regex **s_shortNameRegex** = new Regex(@"", (Version|Culture|PublicKeyToken)=[^\, \]]+", RegexOptions.Compiled)

Properties

- bool **IsValid** [get]
Is the type valid.
- Type **Value** [get]
Retrieve the type. The value is obtained using Type.GetType(string) and cached in a static

6.306.1 Detailed Description

A System.Type wrapper that can be serialized.

A generic version of [SerializableType](#) that can be used to store types that inherit from a specific base type.

Template Parameters

<i>BaseType</i>	The base type of the type stored
-----------------	----------------------------------

6.306.2 Constructor & Destructor Documentation

6.306.2.1 SerializableType() [1/2]

```
SerializableType (
    Type type )
```

Create a new instance and stores full Type.AssemblyQualifiedName. To use shorter form, use [AsShort](#).

Parameters

<i>type</i>	Type to store. Can be null.
-------------	-----------------------------

6.306.2.2 SerializableType() [2/2]

```
SerializableType (
    string type )
```

Create a new instance and stores *type* as [AssemblyQualifiedName](#).

Parameters

<i>type</i>	Type name.
-------------	------------

6.306.3 Member Function Documentation

6.306.3.1 AsShort()

```
SerializableType AsShort ( )
```

Converts [AssemblyQualifiedName](#) and returns a short form, without version, culture etc.

6.306.3.2 Equals() [1/2]

```
override bool Equals (
    object obj )
```

Returns `true` if *obj* is [SerializableType](#) and the [AssemblyQualifiedName](#) is the same.

6.306.3.3 Equals() [2/2]

```
bool Equals (
    SerializableType< BaseType > other )
```

Returns `true` if the [AssemblyQualifiedName](#) is the same.

6.306.3.4 GetHashCode()

```
override int GetHashCode ( )
```

Returns the hash code of the [AssemblyQualifiedName](#).

6.306.3.5 GetShortAssemblyQualifiedName()

```
static string GetShortAssemblyQualifiedName (
    Type type ) [static]
```

Converts the `Type.AssemblyQualifiedName` to a shorter form, without version, culture etc.

6.306.3.6 operator SerializableType()

```
static implicit operator SerializableType (  
    Type type ) [static]
```

Implicitly convert a Type to a [SerializableType](#).

6.306.3.7 operator Type()

```
static implicit operator Type (  
    SerializableType< BaseType > serializableType ) [static]
```

Implicitly convert a [SerializableType](#) to a Type.

6.306.4 Member Data Documentation

6.306.4.1 AssemblyQualifiedName

```
string AssemblyQualifiedName
```

Type's assembly qualified name.

6.306.5 Property Documentation

6.306.5.1 IsValid

```
bool IsValid [get]
```

Is the type valid.

6.306.5.2 Value

```
Type Value [get]
```

Retrieve the type. The value is obtained using `Type.GetType(string)` and cached in a static

6.307 SerializableTypeAttribute Class Reference

Specifies that either a string field represents a type name or sets additional options for [SerializableType](#) field.

Inherits [PropertyAttribute](#).

Properties

- Type [BaseType](#) [get, set]
The base type of the picked type.
- bool [UseFullAssemblyQualifiedName](#) [get, set]
Should the type be stored as a full assembly qualified name.
- bool [WarnIfNoPreserveAttribute](#) [get, set]
Should a warning be shown if the field does not have a PreserveAttribute.

6.307.1 Detailed Description

Specifies that either a string field represents a type name or sets additional options for [SerializableType](#) field.

6.307.2 Property Documentation

6.307.2.1 BaseType

```
Type BaseType [get], [set]
```

The base type of the picked type.

6.307.2.2 UseFullAssemblyQualifiedName

```
bool UseFullAssemblyQualifiedName [get], [set]
```

Should the type be stored as a full assembly qualified name.

6.307.2.3 WarnIfNoPreserveAttribute

```
bool WarnIfNoPreserveAttribute [get], [set]
```

Should a warning be shown if the field does not have a PreserveAttribute.

6.308 SerializeReferenceTypePickerAttribute Class Reference

Attribute used to show a type picker for a field with [SerializeReference].

Inherits [DecoratingPropertyAttribute](#).

Public Member Functions

- [SerializeReferenceTypePickerAttribute](#) (params Type[] types)
Initializes a new instance of the [SerializeReferenceTypePickerAttribute](#) class.

Public Attributes

- bool [GroupTypesByNamespace](#) = true
Should the types be grouped by namespace?
- bool [ShowFullName](#) = false
Should the full name be shown?

Properties

- Type[] [Types](#) [get]
Gets the types to be picked.

Additional Inherited Members

6.308.1 Detailed Description

Attribute used to show a type picker for a field with [SerializeReference].

6.308.2 Constructor & Destructor Documentation

6.308.2.1 SerializeReferenceTypePickerAttribute()

```
SerializeReferenceTypePickerAttribute (  
    params Type[] types )
```

Initializes a new instance of the [SerializeReferenceTypePickerAttribute](#) class.

Parameters

<i>types</i>	The types to be picked.
--------------	-------------------------

6.308.3 Member Data Documentation

6.308.3.1 GroupTypesByNamespace

```
bool GroupTypesByNamespace = true
```

Should the types be grouped by namespace?

6.308.3.2 ShowFullName

```
bool ShowFullName = false
```

Should the full name be shown?

6.308.4 Property Documentation

6.308.4.1 Types

```
Type [] Types [get]
```

Gets the types to be picked.

6.309 SessionInfo Class Reference

Holds information about the Game Session

Public Member Functions

- override string [ToString](#) ()
String representation of a [SessionInfo](#)
- bool [UpdateCustomProperties](#) (Dictionary< string, SessionProperty > customProperties)
Update or change the Custom Properties of the current joined Room

Static Public Member Functions

- static implicit [operator bool](#) ([SessionInfo](#) sessionInfo)
Check if the [SessionInfo](#) reference is not Null and is Valid.

Properties

- `bool?? IsOpen` [get, set]
Signal if the current connected Room is open
- `bool IsValid` [get]
Flag to signal if the [SessionInfo](#) is ready for use
- `bool?? IsVisible` [get, set]
Signal if the current connected Room is visible
- `int MaxPlayers` [get]
Max number of peer that can join this Session, this value always include an extra slot for the Server/Host
- `string Name` [get]
Stores the current Room Name
- `int PlayerCount` [get]
Current number of peers inside this Session, this includes the Server/Host and Clients
- `ReadOnlyDictionary< string, SessionProperty > Properties` [get]
Room Custom Properties
- `string Region` [get]
Stores the current connected Region

6.309.1 Detailed Description

Holds information about the Game Session

6.309.2 Member Function Documentation

6.309.2.1 operator bool()

```
static implicit operator bool (
    SessionInfo sessionInfo ) [static]
```

Check if the [SessionInfo](#) reference is not Null and is Valid.

Parameters

<code>sessionInfo</code>	Session Info
--------------------------	--------------

6.309.2.2 ToString()

```
override string ToString ( )
```

String representation of a [SessionInfo](#)

Returns

Formatted [SessionInfo](#)

6.309.2.3 UpdateCustomProperties()

```
bool UpdateCustomProperties (
    Dictionary< string, SessionProperty > customProperties )
```

Update or change the Custom Properties of the current joined Room

Parameters

<i>customProperties</i>	New custom properties
-------------------------	-----------------------

6.309.3 Property Documentation**6.309.3.1 IsOpen**

```
bool?? IsOpen [get], [set]
```

Signal if the current connected Room is open

6.309.3.2 IsValid

```
bool IsValid [get]
```

Flag to signal if the [SessionInfo](#) is ready for use

6.309.3.3 IsVisible

```
bool?? IsVisible [get], [set]
```

Signal if the current connected Room is visible

6.309.3.4 MaxPlayers

```
int MaxPlayers [get]
```

Max number of peer that can join this Session, this value always include an extra slot for the Server/Host

6.309.3.5 Name

```
string Name [get]
```

Stores the current Room Name

6.309.3.6 PlayerCount

```
int PlayerCount [get]
```

Current number of peers inside this Session, this includes the Server/Host and Clients

6.309.3.7 Properties

```
ReadOnlyDictionary<string, SessionProperty> Properties [get]
```

Room Custom Properties

6.309.3.8 Region

```
string Region [get]
```

Stores the current connected Region

6.310 Simulation Class Reference

Main simulation class

Inherits [ILogSource](#), and [INetPeerGroupCallbacks](#).

Classes

- struct [AreaOfInterest](#)
Area of Interest Definition

Public Member Functions

- void [GetAreaOfInterestGizmoData](#) (List<(Vector3 center, Vector3 size, int playerCount, int objectCount)> result)

Clears the provided list and populates it with data about the current Area of Interest (AOI) cells. Each tuple in the list represents one AOI cell, containing its center, size, player count, and object count.
- [PlayerRef](#) [GetInputAuthority](#) ([NetworkObject](#) networkObject)

Get the Input Authority [PlayerRef](#) for a [NetworkObject](#)
- [SimulationInput](#) [GetInputForPlayer](#) ([PlayerRef](#) player)

Get the [Simulation](#) Input for a specific [Player](#)
- void [GetObjectsAndPlayersInAreaOfInterestCell](#) (int cellKey, List< [PlayerRef](#) > players, List< [NetworkId](#) > objects)

Used by [RunnerAOIGizmos](#) component. Supplies data about current active AOI cells.
- List< [NetworkId](#) > [GetObjectsInAreaOfInterestForPlayer](#) ([PlayerRef](#) player)

Retrieves a list of network object IDs that are in the area of interest for the specified player.
- [PlayerRef](#) [GetStateAuthority](#) ([NetworkObject](#) networkObject)

Get the State Authority [PlayerRef](#) for a [NetworkObject](#)
- bool [HasAnyActiveConnections](#) ()

Signal if the Server has any Active Connection with any number of Clients.
- bool [IsInputAuthority](#) ([NetworkObject](#) networkObject, [PlayerRef](#) playerRef)

Check if a [Player](#) is the Input Authority over an [NetworkObject](#)
- bool? [IsInterestedIn](#) ([NetworkObject](#) obj, [PlayerRef](#) player)

Check if a [NetworkObject](#) is interested by a specific [Player](#)
- bool [IsLocalSimulationInputAuthority](#) ([NetworkObject](#) obj)

Check if the Local [Player](#) is the Input Authority over a [NetworkObject](#)
- bool [IsLocalSimulationStateAuthority](#) ([NetworkId](#) id)

Check if a [Player](#) is the State Authority over a [NetworkObject](#) by [NetworkId](#)
- bool [IsLocalSimulationStateAuthority](#) ([NetworkObject](#) obj)

Check if the Local [Player](#) is the State Authority over a [NetworkObject](#)
- bool [IsLocalSimulationStateOrInputSource](#) ([NetworkObject](#) obj)

Check if the Local [Player](#) is the State Authority or Input Authority over a [NetworkObject](#)
- bool [IsStateAuthority](#) ([NetworkObject](#) networkObject, [PlayerRef](#) playerRef)

Check if a [Player](#) is the State Authority over a [NetworkObject](#)
- bool [IsStateAuthority](#) ([PlayerRef](#) stateSource, [PlayerRef](#) playerRef)

Check if a [Player](#) is the State Authority in relation to another [Player](#)
- bool [TryGetHostPlayer](#) (out [PlayerRef](#) player)

Try to get the Host [Player](#)
- int [Update](#) (double dt)

Forwards the [Simulation](#) based on the Delta Time

Protected Member Functions

- virtual void [AfterSimulation](#) ()

Callback invoked after the [Simulation](#) Update
- virtual void [AfterUpdate](#) ()

Callback invoked After the [Simulation](#) Update
- virtual void [BeforeFirstTick](#) ()

Callback invoked before the First [Tick](#)
- virtual int [BeforeSimulation](#) ()

Callback invoked before the [Simulation](#) Loop
- virtual void [BeforeUpdate](#) ()

- Callback invoked before the [Simulation Update](#)*

 - virtual void [NetworkConnected](#) ([NetConnection](#) *connection)

Callback invoked on the Connected
- virtual void [NetworkDisconnected](#) ([NetConnection](#) *connection, [NetDisconnectReason](#) reason)

Callback invoked on the Disconnected
- virtual void [NetworkReceiveDone](#) ()

Callback invoked when the Network Receive is completed
- virtual void [NoSimulation](#) ()

Callback invoked when there is no simulation

Properties

- virtual IEnumerable< [PlayerRef](#) > [ActivePlayers](#) [get]

List of Active players in the [Simulation](#)
- [SimulationConfig](#) [Config](#) [get]

The [SimulationConfig](#) file used by this [Simulation](#).
- float [DeltaTime](#) [get]

Gets the fixed tick time interval. Derived from the [SimulationRuntimeConfig.TickRate](#).
- int [InputCount](#) [get]

The current input collection size
- bool [IsClient](#) [get]

If this peer is a client. True for client peers in Server/Client topologies, and true for all peers in Shared Mode.
- bool [IsFirstTick](#) [get]

Use in conjunction with [IsResimulation/IsForward](#) inside of [FixedUpdateNetwork](#) to determine if the current tick being simulated is the first tick of the resimulation or forward phase of the simulation loop.
- bool [IsForward](#) [get]

Use inside of [FixedUpdateNetwork](#) to determine if the tick currently being simulated has NOT previously been simulated locally.
- bool [IsLastTick](#) [get]

Use in conjunction with [IsResimulation/IsForward](#) inside of [FixedUpdateNetwork](#) to determine if the current tick being simulated is the last tick of the resimulation or forward phase of the simulation loop.
- bool [IsLocalPlayerFirstExecution](#) [get]

True if the current stage of the simulation loop is Forward. False during resimulations.
- bool [IsMasterClient](#) [get]

Only valid in Shared Mode. Indicates if this peer is flagged as the MasterClient, which means it is default State↔ Authority
- bool [IsPlayer](#) [get]

True for any peer that represents a human player. This is true for all peers except a dedicated server.
- bool [IsResimulation](#) [get]

Use inside of [FixedUpdateNetwork](#) to determine if the tick currently being simulated has previously been simulated locally. Resimulation occurs in client prediction when new states arrive from the StateAuthority. Networked objects are set to the most current authority state tick, and simulations are repeated from that tick to the local current tick.
- bool [IsRunning](#) [get]

Signal if the [Simulation](#) is currently running
- bool [IsServer](#) [get]

If this peer is the server. True for the Server or Host peer in Server/Client topologies, and always false for all peers in Shared Mode (the relay is the server).
- bool [IsShutdown](#) [get]
- bool [IsSinglePlayer](#) [get]

Indicates that this simulation is operating in Single Player mode, which is a Host that accepts no connections.
- abstract [Tick LatestServerTick](#) [get]

- latest tick on server we are aware of*

 - [NetAddress LocalAddress](#) [get]

Bound Address of the internal socket
- float [LocalAlpha](#) [get]
- abstract [PlayerRef LocalPlayer](#) [get]

Get LocalPlayer PlayerRef
- [SimulationModes Mode](#) [get]

Gets the SimulationModes flags for The type of network peer this simulation represents.
- [NetConfig * NetConfigPointer](#) [get]

Current NetConfig
- int [ObjectCount](#) [get]

Returns the number of objects in the simulation.
- Dictionary< [NetworkId, NetworkObjectMeta](#) > [Objects](#) [get]

Returns a map of all objects in the simulation.
- [NetworkProjectConfig ProjectConfig](#) [get]

The NetworkProjectConfig file used by this Simulation.
- float [RemoteAlpha](#) [get]

Remote Interpolation Alpha
- [Tick RemoteTick](#) [get]

Remote Tick
- [Tick RemoteTickPrevious](#) [get]

Remote previous Tick
- double [SendDelta](#) [get]

The packet send delta time
- int [SendRate](#) [get]

The packet send rate
- [SimulationStages Stage](#) [get]

Gets the current SimulationStages value.
- [Tick Tick](#) [get]

The tick associated with the current state of networked objects, or the current simulation tick being processed (when evaluated during FixedUpdateNetwork).
- double [TickDeltaDouble](#) [get]

The delta time of each tick as a double
- float [TickDeltaFloat](#) [get]

The delta time of each tick as a float
- [Tick TickPrevious](#) [get]

The previous tick
- int [TickRate](#) [get]

The current tick rate of the simulation
- int [TickStride](#) [get]

How large the ticks the current simulation takes are
- double [Time](#) [get]

The current simulation time in seconds
- [Topologies Topology](#) [get]

Indicates if a Server/Client or Shared Mode (relay server) topology is being used.

6.310.1 Detailed Description

Main simulation class

6.310.2 Member Function Documentation

6.310.2.1 AfterSimulation()

```
virtual void AfterSimulation ( ) [protected], [virtual]
```

Callback invoked after the [Simulation](#) Update

6.310.2.2 AfterUpdate()

```
virtual void AfterUpdate ( ) [protected], [virtual]
```

Callback invoked After the [Simulation](#) Update

6.310.2.3 BeforeFirstTick()

```
virtual void BeforeFirstTick ( ) [protected], [virtual]
```

Callback invoked before the First [Tick](#)

6.310.2.4 BeforeSimulation()

```
virtual int BeforeSimulation ( ) [protected], [virtual]
```

Callback invoked before the [Simulation](#) Loop

Returns

Total number of re-simulations

6.310.2.5 BeforeUpdate()

```
virtual void BeforeUpdate ( ) [protected], [virtual]
```

Callback invoked before the [Simulation](#) Update

6.310.2.6 GetAreaOfInterestGizmoData()

```
void GetAreaOfInterestGizmoData (
    List<(Vector3 center, Vector3 size, int playerCount, int objectCount)> result )
```

Clears the provided list and populates it with data about the current Area of Interest (AOI) cells. Each tuple in the list represents one AOI cell, containing its center, size, player count, and object count.

Parameters

<i>result</i>	The list to be populated with AOI cell data.
---------------	--

6.310.2.7 GetInputAuthority()

```
PlayerRef GetInputAuthority (
    NetworkObject networkObject )
```

Get the Input Authority [PlayerRef](#) for a [NetworkObject](#)

Parameters

<i>networkObject</i>	NetworkObject to check
----------------------	--

Returns

[PlayerRef](#) of the Input Authority for the [NetworkObject](#)

6.310.2.8 GetInputForPlayer()

```
SimulationInput GetInputForPlayer (
    PlayerRef player )
```

Get the [Simulation](#) Input for a specific Player

Parameters

<i>player</i>	Player to check for the Simulation Input
---------------	--

Returns

[Simulation](#) Input for a specific Player

6.310.2.9 GetObjectsAndPlayersInAreaOfInterestCell()

```
void GetObjectsAndPlayersInAreaOfInterestCell (
    int cellKey,
    List< PlayerRef > players,
    List< NetworkId > objects )
```

Used by RunnerAOIGizmos component. Supplies data about current active AOI cells.

6.310.2.10 GetObjectsInAreaOfInterestForPlayer()

```
List<NetworkId> GetObjectsInAreaOfInterestForPlayer (
    PlayerRef player )
```

Retrieves a list of network object IDs that are in the area of interest for the specified player.

Parameters

<i>player</i>	The player for whom the area of interest is being queried.
---------------	--

Returns

A list of network object IDs in the area of interest for the player.

6.310.2.11 GetStateAuthority()

```
PlayerRef GetStateAuthority (
    NetworkObject networkObject )
```

Get the State Authority [PlayerRef](#) for a [NetworkObject](#)

Parameters

<i>networkObject</i>	NetworkObject to check
----------------------	--

Returns

[PlayerRef](#) of the State Authority for the [NetworkObject](#)

6.310.2.12 HasAnyActiveConnections()

```
bool HasAnyActiveConnections ( )
```

Signal if the Server has any Active Connection with any number of Clients.

Returns

True, if at least one connection is active, false otherwise.

6.310.2.13 IsInputAuthority()

```
bool IsInputAuthority (
    NetworkObject networkObject,
    PlayerRef playerRef )
```

Check if a Player is the Input Authority over an [NetworkObject](#)

Parameters

<i>networkObject</i>	NetworkObject to check
<i>playerRef</i>	Player to check

Returns

True if the Player is the Input Authority over the [NetworkObject](#), false otherwise

6.310.2.14 IsInterestedIn()

```
bool? IsInterestedIn (
    NetworkObject obj,
    PlayerRef player )
```

Check if a [NetworkObject](#) is interested by a specific Player

Parameters

<i>obj</i>	NetworkObject to check
<i>player</i>	Player to check

Returns

True if the Player is interested in the [NetworkObject](#), false otherwise

6.310.2.15 IsLocalSimulationInputAuthority()

```
bool IsLocalSimulationInputAuthority (
    NetworkObject obj )
```

Check if the Local Player is the Input Authority over a [NetworkObject](#)

Parameters

<i>obj</i>	NetworkObject to check
------------	--

Returns

True if the Player is the Input Authority, false otherwise

6.310.2.16 IsLocalSimulationStateAuthority() [1/2]

```
bool IsLocalSimulationStateAuthority (
    NetworkId id )
```

Check if a Player is the State Authority over a [NetworkObject](#) by [NetworkId](#)

Parameters

<i>id</i>	NetworkId of the NetworkObject to check
-----------	---

Returns

True if the Player is the State Authority, false otherwise

6.310.2.17 IsLocalSimulationStateAuthority() [2/2]

```
bool IsLocalSimulationStateAuthority (
    NetworkObject obj )
```

Check if the Local Player is the State Authority over a [NetworkObject](#)

Parameters

<i>obj</i>	NetworkObject to check
------------	--

Returns

True if the Player is the State Authority, false otherwise

6.310.2.18 IsLocalSimulationStateOrInputSource()

```
bool IsLocalSimulationStateOrInputSource (
    NetworkObject obj )
```

Check if the Local Player is the State Authority or Input Authority over a [NetworkObject](#)

Parameters

<i>obj</i>	NetworkObject to check
------------	--

Returns

True if the Player is the State Authority or Input Authority, false otherwise

6.310.2.19 IsStateAuthority() [1/2]

```
bool IsStateAuthority (
    NetworkObject networkObject,
    PlayerRef playerRef )
```

Check if a Player is the State Authority over a [NetworkObject](#)

Parameters

<i>networkObject</i>	NetworkObject to check
<i>playerRef</i>	Player to check

Returns

True if the Player is the State Authority, false otherwise

6.310.2.20 IsStateAuthority() [2/2]

```
bool IsStateAuthority (
    PlayerRef stateSource,
    PlayerRef playerRef )
```

Check if a Player is the State Authority in relation to another Player

Parameters

<i>stateSource</i>	State Source Player
<i>playerRef</i>	Player to check

Returns

True if the Player is the State Authority, false otherwise

6.310.2.21 NetworkConnected()

```
virtual void NetworkConnected (
    NetConnection * connection ) [protected], [virtual]
```

Callback invoked on the Connected

Parameters

<i>connection</i>	Connection that was connected
-------------------	-------------------------------

6.310.2.22 NetworkDisconnected()

```
virtual void NetworkDisconnected (
    NetConnection * connection,
    NetDisconnectReason reason ) [protected], [virtual]
```

Callback invoked on the Disconnected

Parameters

<i>connection</i>	Connection that was disconnected
<i>reason</i>	Reason for the disconnection

6.310.2.23 NetworkReceiveDone()

```
virtual void NetworkReceiveDone ( ) [protected], [virtual]
```

Callback invoked when the Network Receive is completed

6.310.2.24 NoSimulation()

```
virtual void NoSimulation ( ) [protected], [virtual]
```

Callback invoked when there is no simulation

6.310.2.25 TryGetHostPlayer()

```
bool TryGetHostPlayer (
    out PlayerRef player )
```

Try to get the Host Player

Parameters

<i>player</i>	Host Player
---------------	-------------

Returns

True if the Host Player was found, false otherwise

6.310.2.26 Update()

```
int Update (
    double dt )
```

Forwards the [Simulation](#) based on the Delta Time

Parameters

<i>dt</i>	Delta Time used to forward the simulation
-----------	---

Returns

How many Ticks executed on this Update

6.310.3 Property Documentation**6.310.3.1 ActivePlayers**

```
virtual IEnumerable<PlayerRef> ActivePlayers [get]
```

List of Active players in the [Simulation](#)

6.310.3.2 Config

```
SimulationConfig Config [get]
```

The [SimulationConfig](#) file used by this [Simulation](#).

6.310.3.3 DeltaTime

```
float DeltaTime [get]
```

Gets the fixed tick time interval. Derived from the [SimulationRuntimeConfig.TickRate](#).

6.310.3.4 InputCount

```
int InputCount [get]
```

The current input collection size

6.310.3.5 IsClient

```
bool IsClient [get]
```

If this peer is a client. True for client peers in Server/Client topologies, and true for all peers in Shared Mode.

6.310.3.6 IsFirstTick

```
bool IsFirstTick [get]
```

Use in conjunction with IsResimulation/IsForward inside of FixedUpdateNetwork to determine if the current tick being simulated is the first tick of the resimulation or forward phase of the simulation loop.

'Resimulation' describes simulating a tick that has been previously been simulated.

'Forward' describes simulating a tick that is being simulated for the first time locally.

'Prediction' describes simulating ticks higher than the most current known StateAuthority snapshot tick.

6.310.3.7 IsForward

```
bool IsForward [get]
```

Use inside of FixedUpdateNetwork to determine if the tick currently being simulated has NOT previously been simulated locally.

6.310.3.8 IsLastTick

```
bool IsLastTick [get]
```

Use in conjunction with IsResimulation/IsForward inside of FixedUpdateNetwork to determine if the current tick being simulated is the last tick of the resimulation or forward phase of the simulation loop.

'Resimulation' describes simulating a tick that has been previously been simulated.

'Forward' describes simulating a tick that is being simulated for the first time locally.

'Prediction' describes simulating ticks higher than the most current known StateAuthority snapshot tick.

6.310.3.9 IsLocalPlayerFirstExecution

```
bool IsLocalPlayerFirstExecution [get]
```

True if the current stage of the simulation loop is Forward. False during resimulations.

'Resimulation' describes simulating a tick that has been previously been simulated.

'Forward' describes simulating a tick that is being simulated for the first time locally.

'Prediction' describes simulating ticks higher than the most current known StateAuthority snapshot tick.

6.310.3.10 IsMasterClient

```
bool IsMasterClient [get]
```

Only valid in Shared Mode. Indicates if this peer is flagged as the MasterClient, which means it is default StateAuthority

6.310.3.11 IsPlayer

```
bool IsPlayer [get]
```

True for any peer that represents a human player. This is true for all peers except a dedicated server.

6.310.3.12 IsResimulation

```
bool IsResimulation [get]
```

Use inside of FixedUpdateNetwork to determine if the tick currently being simulated has previously been simulated locally. Resimulation occurs in client prediction when new states arrive from the StateAuthority. Networked objects are set to the most current authority state tick, and simulations are repeated from that tick to the local current tick.

6.310.3.13 IsRunning

```
bool IsRunning [get]
```

Signal if the [Simulation](#) is currently running

6.310.3.14 IsServer

```
bool IsServer [get]
```

If this peer is the server. True for the Server or Host peer in Server/Client topologies, and always false for all peers in Shared Mode (the relay is the server).

6.310.3.15 IsShutdown

```
bool IsShutdown [get]
```

6.310.3.16 IsSinglePlayer

```
bool IsSinglePlayer [get]
```

Indicates that this simulation is operating in Single Player mode, which is a Host that accepts no connections.

6.310.3.17 LatestServerTick

```
abstract Tick LatestServerTick [get]
```

latest tick on server we are aware of

6.310.3.18 LocalAddress

```
NetAddress LocalAddress [get]
```

Bound Address of the internal socket

6.310.3.19 LocalAlpha

```
float LocalAlpha [get]
```

6.310.3.20 LocalPlayer

```
abstract PlayerRef LocalPlayer [get]
```

Get LocalPlayer [PlayerRef](#)

6.310.3.21 Mode

`SimulationModes Mode [get]`

Gets the [SimulationModes](#) flags for The type of network peer this simulation represents.

6.310.3.22 NetConfigPointer

`NetConfig* NetConfigPointer [get]`

Current NetConfig

6.310.3.23 ObjectCount

`int ObjectCount [get]`

Returns the number of objects in the simulation.

6.310.3.24 Objects

`Dictionary<NetworkId, NetworkObjectMeta> Objects [get]`

Returns a map of all objects in the simulation.

6.310.3.25 ProjectConfig

`NetworkProjectConfig ProjectConfig [get]`

The [NetworkProjectConfig](#) file used by this [Simulation](#).

6.310.3.26 RemoteAlpha

`float RemoteAlpha [get]`

Remote Interpolation Alpha

6.310.3.27 RemoteTick

`Tick RemoteTick [get]`

Remote [Tick](#)

6.310.3.28 RemoteTickPrevious

`Tick RemoteTickPrevious [get]`

Remote previous [Tick](#)

6.310.3.29 SendDelta

`double SendDelta [get]`

The packet send delta time

6.310.3.30 SendRate

`int SendRate [get]`

The packet send rate

6.310.3.31 Stage

`SimulationStages Stage [get]`

Gets the current [SimulationStages](#) value.

6.310.3.32 Tick

`Tick Tick [get]`

The tick associated with the current state of networked objects, or the current simulation tick being processed (when evaluated during `FixedUpdateNetwork`).

6.310.3.33 TickDeltaDouble

```
double TickDeltaDouble [get]
```

The delta time of each tick as a double

6.310.3.34 TickDeltaFloat

```
float TickDeltaFloat [get]
```

The delta time of each tick as a float

6.310.3.35 TickPrevious

```
Tick TickPrevious [get]
```

The previous tick

6.310.3.36 TickRate

```
int TickRate [get]
```

The current tick rate of the simulation

6.310.3.37 TickStride

```
int TickStride [get]
```

How large the ticks the current simulation takes are

6.310.3.38 Time

```
double Time [get]
```

The current simulation time in seconds

6.310.3.39 Topology

[Topologies](#) `Topology` [get]

Indicates if a Server/Client or Shared Mode (relay server) topology is being used.

6.311 Simulation.AreaOfInterest Struct Reference

Area of Interest Definition

Static Public Member Functions

- static int int int z **ClampCellCoords** (int x, int y, int z)
- static int [GetCellSize](#) ()
 - Get the size of each cell in the AOI grid.*
- static int int int z **GetGridSize** ()
- static void [SphereToCells](#) (Vector3 position, float radius, HashSet< int > cells)
 - Convert a sphere into a set of AOI cells.*
- static int [ToCell](#) (int x, int y, int z)
 - Convert a cell coordinate into the respective cell index.*
- static int [ToCell](#) (Vector3 position)
 - Convert a position into the respective cell index.*
- static Vector3 [ToCellCenter](#) (int index)
 - Convert a cell index into the respective cell center position.*
- static int int int z **ToCellCoords** (int index)
- static int int int z **ToCellCoords** (Vector3 position)

Static Public Attributes

- static int [CELL_SIZE](#) = SIZE_DEFAULT
 - Size of each cell in the AOI grid.*
- static int [x](#)
 - Get the size of the AOI grid.*
- static int int [y](#)

6.311.1 Detailed Description

Area of Interest Definition

6.311.2 Member Function Documentation

6.311.2.1 GetCellSize()

```
static int GetCellSize ( ) [static]
```

Get the size of each cell in the AOI grid.

Returns

The size of each cell in the AOI grid.

6.311.2.2 SphereToCells()

```
static void SphereToCells (
    Vector3 position,
    float radius,
    HashSet< int > cells ) [static]
```

Convert a sphere into a set of AOI cells.

Parameters

<i>position</i>	Sphere center
<i>radius</i>	Sphere radius. Max allowed radius is MAX_SHARED_RADIUS
<i>cells</i>	Resulting set of cells

6.311.2.3 ToCell() [1/2]

```
static int ToCell (
    int x,
    int y,
    int z ) [static]
```

Convert a cell coordinate into the respective cell index.

Parameters

<i>x</i>	X coordinate
<i>y</i>	Y coordinate
<i>z</i>	Z coordinate

Returns

Cell index

6.311.2.4 ToCell() [2/2]

```
static int ToCell (  
    Vector3 position ) [static]
```

Convert a position into the respective cell index.

Parameters

<i>position</i>	Position
-----------------	----------

Returns

Cell index

6.311.2.5 ToCellCenter()

```
static Vector3 ToCellCenter (  
    int index ) [static]
```

Convert a cell index into the respective cell center position.

Parameters

<i>index</i>	Cell index
--------------	------------

Returns

Cell center position

6.311.3 Member Data Documentation

6.311.3.1 CELL_SIZE

```
int CELL_SIZE = SIZE_DEFAULT [static]
```

Size of each cell in the AOI grid.

6.311.3.2 x

```
static int x [static]
```

Get the size of the AOI grid.

Clamp cell coordinates to the valid range.

Converts a cell index into its corresponding cell coordinates.

Convert a position into the respective cell coordinate.

Returns

The size of the AOI grid.

Parameters

<i>position</i>	Position
-----------------	----------

Returns

Cell coordinate

Parameters

<i>index</i>	The index of the cell to be converted.
--------------	--

Returns

A tuple containing the x, y, and z coordinates of the cell.

Parameters

<i>x</i>	X coordinate
<i>y</i>	Y coordinate
<i>z</i>	Z coordinate

Returns

Clamped cell coordinates

6.312 SimulationBehaviour Class Reference

Base class for a [Fusion](#) aware [Behaviour](#) (derived from `UnityEngine.MonoBehaviour`). If a [SimulationBehaviour](#) is found on a [NetworkRunner](#) game object during the runner initialisation, the [SimulationBehaviour](#) is automatically registered. Objects derived from this object can be associated with a [NetworkRunner](#) and [Simulation](#) using [NetworkRunner.AddGlobal\(\)](#).

Inherits [Behaviour](#).

Inherited by [HitboxManager](#), and [NetworkBehaviour](#).

Public Member Functions

- virtual void [FixedUpdateNetwork](#) ()
Fusion FixedUpdate timing callback.
- virtual void [Render](#) ()
Post simulation frame rendering callback. Runs after all simulations have finished. Use in place of Unity's Update when Fusion is handling Physics.

Properties

- bool [CanReceiveRenderCallback](#) [get]
Gets a value indicating whether this instance can receive render callbacks.
- bool [CanReceiveSimulationCallback](#) [get]
Gets a value indicating whether this instance can receive simulation callbacks.
- [NetworkObject](#) [Object](#) [get]
The NetworkObject this component is associated with.
- [NetworkRunner](#) [Runner](#) [get]
The NetworkRunner this component is associated with.

Additional Inherited Members

6.312.1 Detailed Description

Base class for a [Fusion](#) aware [Behaviour](#) (derived from `UnityEngine.MonoBehaviour`). If a [SimulationBehaviour](#) is found on a [NetworkRunner](#) game object during the runner initialisation, the [SimulationBehaviour](#) is automatically registered. Objects derived from this object can be associated with a [NetworkRunner](#) and [Simulation](#) using [NetworkRunner.AddGlobal\(\)](#).

6.312.2 Member Function Documentation

6.312.2.1 FixedUpdateNetwork()

```
virtual void FixedUpdateNetwork ( ) [virtual]
```

[Fusion](#) FixedUpdate timing callback.

Reimplemented in [NetworkBehaviour](#).

6.312.2.2 Render()

```
virtual void Render ( ) [virtual]
```

Post simulation frame rendering callback. Runs after all simulations have finished. Use in place of Unity's Update when [Fusion](#) is handling Physics.

Reimplemented in [NetworkTransform](#), and [NetworkMecanimAnimator](#).

6.312.3 Property Documentation

6.312.3.1 CanReceiveRenderCallback

```
bool CanReceiveRenderCallback [get]
```

Gets a value indicating whether this instance can receive render callbacks.

`true` if this instance can receive render callbacks; otherwise, `false`.

This property checks the current flags of the instance against various conditions to determine if it can receive render callbacks.

6.312.3.2 CanReceiveSimulationCallback

```
bool CanReceiveSimulationCallback [get]
```

Gets a value indicating whether this instance can receive simulation callbacks.

`true` if this instance can receive simulation callbacks; otherwise, `false`.

This property checks the current flags of the instance against various conditions to determine if it can receive simulation callbacks.

6.312.3.3 Object

```
NetworkObject Object [get]
```

The [NetworkObject](#) this component is associated with.

6.312.3.4 Runner

```
NetworkRunner Runner [get]
```

The [NetworkRunner](#) this component is associated with.

6.313 SimulationBehaviourAttribute Class Reference

Attribute for specifying which [SimulationStages](#) and [SimulationModes](#) this [SimulationBehaviour](#) will execute in. Can be used to limit execution to only Host, Server or Client peers, or to only execute on Resimulation or Forward ticks.
Usage:

Inherits [Attribute](#).

Properties

- [SimulationModes Modes](#) [get, set]
Flag for which indicated peers in [SimulationModes](#) will execute this script.
- [SimulationStages Stages](#) [get, set]
Flag for which stages of the simulation loop this component will execute this script.
- [Topologies Topologies](#) [get, set]
Flag for which topologies this script will execute in

6.313.1 Detailed Description

Attribute for specifying which [SimulationStages](#) and [SimulationModes](#) this [SimulationBehaviour](#) will execute in. Can be used to limit execution to only Host, Server or Client peers, or to only execute on Resimulation or Forward ticks. Usage:

```
[SimulationBehaviour(Stages = SimulationStages.Forward, Modes = SimulationModes.Server
| SimulationModes.Host)]
```

6.313.2 Property Documentation

6.313.2.1 Modes

```
SimulationModes Modes [get], [set]
```

Flag for which indicated peers in [SimulationModes](#) will execute this script.

6.313.2.2 Stages

```
SimulationStages Stages [get], [set]
```

Flag for which stages of the simulation loop this component will execute this script.

6.313.2.3 Topologies

```
Topologies Topologies [get], [set]
```

Flag for which topologies this script will execute in

6.314 SimulationBehaviourListScope Struct Reference

Provides a scope for a `SimulationBehaviourUpdater.BehaviourList`, incrementing its lock count on creation and decrementing it on disposal. If the lock count reaches zero on disposal, all pending removals in the list are performed.

Inherits `IDisposable`.

Public Member Functions

- void `Dispose` ()
Dispose unmanaged resources.

6.314.1 Detailed Description

Provides a scope for a `SimulationBehaviourUpdater.BehaviourList`, incrementing its lock count on creation and decrementing it on disposal. If the lock count reaches zero on disposal, all pending removals in the list are performed.

6.314.2 Member Function Documentation

6.314.2.1 `Dispose()`

```
void Dispose ( )
```

Dispose unmanaged resources.

6.315 SimulationConfig Class Reference

Project configuration settings specific to how the `Simulation` class behaves.

Inherits `IConfigurationSanityCheck`.

Public Types

- enum class `DataConsistency`
- enum class `InputTransferModes`
- enum class `SimulationTimeMode`

The time mode that the `NetworkRunnerUpdaterDefault` uses to calculate the delta time for the simulation update.

Public Member Functions

- void `SanityCheck` ()

Public Attributes

- bool [HostMigration](#)
If, in host mode, we should allow host migration if the current host leaves.
- int [InputDataWordCount](#)
Input Data Word Count
- [InputTransferModes](#) [InputTransferMode](#)
The way which input is transferred
- [DataConsistency](#) [ObjectDataConsistency](#)
How the server chooses to send [NetworkObject](#) updates to balance consistency and bandwidth consumption.
- int [PlayerCount](#) = 10
The default number of players allowed to join a game instance. Can also be changed in code when starting [Fusion](#).
- [NetworkProjectConfig.ReplicationFeatures](#) [ReplicationFeatures](#) = [NetworkProjectConfig.ReplicationFeatures.Scheduling](#)
Scheduling is the default.
- [SimulationTimeMode](#) [SimulationUpdateTimeMode](#) = [SimulationTimeMode.UnscaledDeltaTime](#)
The time mode that the Runner uses to update the simulation.
- [TickRate.Selection](#) [TickRateSelection](#) = [TickRate.Default](#)
The default tick rate to use. Can also be changed in code when starting [Fusion](#).
- [Topologies](#) [Topology](#)
The topology used

Properties

- bool [AreaOfInterestEnabled](#) [get]
Signal if AOI is enabled
- int [InputTotalWordCount](#) [get]
- bool [SchedulingEnabled](#) [get]
Signal if scheduling is enabled
- bool [SchedulingWithoutAOI](#) [get]
Signal if scheduling is running without AOI

6.315.1 Detailed Description

Project configuration settings specific to how the [Simulation](#) class behaves.

6.315.2 Member Enumeration Documentation

6.315.2.1 DataConsistency

```
enum DataConsistency [strong]
```

Enumerator

Full	When a NetworkBehaviour 's data changes, the server will send all properties whose changes have not been acknowledged. This option consumes more bandwidth, but guarantees that each NetworkBehaviour has consistent state.
Eventual	When a NetworkBehaviour 's data changes, the server will only send the newly changed properties. This option consumes less bandwidth, but a NetworkBehaviour may have inconsistent state at times (some properties up-to-date but not others).

6.315.2.2 InputTransferModes

```
enum InputTransferModes [strong]
```

Enumerator

Redundancy	Send delta compressed and redundant input, used for most games
RedundancyUncompressed	Send redundant input, use for games with small input structs (like fps games) and high player count
LatestState	Only send latest input state, useful for VR, etc.

6.315.2.3 SimulationTimeMode

```
enum SimulationTimeMode [strong]
```

The time mode that the [NetworkRunnerUpdaterDefault](#) uses to calculate the delta time for the simulation update.

Enumerator

UnscaledDeltaTime	Use <code>Time.UnscaledDeltaTime</code> . Time is not affected by timescale and keeps running when the editor is paused.
DeltaTime	Use <code>Time.DeltaTime</code> . Only for GameMode.Single . Can be used to allow for timescale changes in gameplay or to pause the game and continue without interruption when stopping at a debug breakpoint.

6.315.3 Member Data Documentation

6.315.3.1 HostMigration

```
bool HostMigration
```

If, in host mode, we should allow host migration if the current host leaves.

6.315.3.2 InputDataWordCount

```
int InputDataWordCount
```

Input Data Word Count

6.315.3.3 InputTransferMode

```
InputTransferModes InputTransferMode
```

The way which input is transferred

6.315.3.4 ObjectDataConsistency

```
DataConsistency ObjectDataConsistency
```

How the server chooses to send [NetworkObject](#) updates to balance consistency and bandwidth consumption.

6.315.3.5 PlayerCount

```
int PlayerCount = 10
```

The default number of players allowed to join a game instance. Can also be changed in code when starting [Fusion](#).

6.315.3.6 ReplicationFeatures

```
NetworkProjectConfig.ReplicationFeatures ReplicationFeatures = NetworkProjectConfig.ReplicationFeatures.Schedu
```

Scheduling is the default.

6.315.3.7 SimulationUpdateTimeMode

```
SimulationTimeMode SimulationUpdateTimeMode = SimulationTimeMode.UnscaledDeltaTime
```

The time mode that the Runner uses to update the simulation.

-

6.315.3.8 TickRateSelection

```
TickRate.Selection TickRateSelection = TickRate.Default
```

The default tick rate to use. Can also be changed in code when starting [Fusion](#).

6.315.3.9 Topology

```
Topologies Topology
```

The topology used

6.315.4 Property Documentation

6.315.4.1 AreaOfInterestEnabled

```
bool AreaOfInterestEnabled [get]
```

Signal if AOI is enabled

6.315.4.2 InputTotalWordCount

```
int InputTotalWordCount [get]
```

6.315.4.3 SchedulingEnabled

```
bool SchedulingEnabled [get]
```

Signal if scheduling is enabled

6.315.4.4 SchedulingWithoutAOI

```
bool SchedulingWithoutAOI [get]
```

Signal if scheduling is running without AOI

6.316 SimulationInput Class Reference

Simulation Input

Classes

- class [Buffer](#)
Buffer for SimulationInputs.

Public Member Functions

- void [Clear](#) (int wordCount)
Clear a total of wordCount words from this input.
- void [CopyFrom](#) ([SimulationInput](#) source, int wordCount)
Copy wordCount words from source to this input.

Properties

- int * [Data](#) [get]
Data for this input.
- [SimulationInputHeader](#) * [Header](#) [get]
Header for this input.
- [PlayerRef](#) [Player](#) [get, set]
Player that owns this input.
- int [Sent](#) [get, set]
Simulation input sent count.

6.316.1 Detailed Description

Simulation Input

6.316.2 Member Function Documentation

6.316.2.1 Clear()

```
void Clear (  
    int wordCount )
```

Clear a total of *wordCount* words from this input.

Parameters

<i>wordCount</i>	Word count to clear.
------------------	----------------------

6.316.2.2 CopyFrom()

```
void CopyFrom (
    SimulationInput source,
    int wordCount )
```

Copy *wordCount* words from *source* to this input.

Parameters

<i>source</i>	Input to copy from.
<i>wordCount</i>	Word count to copy.

6.316.3 Property Documentation

6.316.3.1 Data

```
int* Data [get]
```

Data for this input.

6.316.3.2 Header

```
SimulationInputHeader* Header [get]
```

Header for this input.

6.316.3.3 Player

```
PlayerRef Player [get], [set]
```

Player that owns this input.

6.316.3.4 Sent

```
int Sent [get], [set]
```

[Simulation](#) input sent count.

6.317 SimulationInput.Buffer Class Reference

[Buffer](#) for [SimulationInputs](#).

Public Member Functions

- bool [Add](#) ([SimulationInput](#) input, double? insertTime=null)
Adds an input to the buffer.
- [Buffer](#) ([NetworkProjectConfig](#) cfg)
Creates a new [Buffer](#).
- void [Clear](#) ()
Clears the buffer.
- bool [Contains](#) ([Tick](#) tick)
Whether the buffer contains an input for tick .
- int [CopySortedTo](#) ([SimulationInput](#)[] array)
Copies the buffer to an array and sorts it.
- [SimulationInput](#) [Get](#) ([Tick](#) tick)
Gets the input for tick .
- double? [GetInsertTime](#) ([Tick](#) tick)
Gets the insert time for tick .
- [SimulationInputHeader](#) [GetLastUsedInputHeader](#) ()
Retrieves the last used input header data.
- bool [Remove](#) ([Tick](#) tick, out [SimulationInput](#) removed)
Removes an input for tick from the buffer.

Properties

- int [Count](#) [get]
Number of inputs in the buffer.
- bool [Full](#) [get]
Whether the buffer is full.

6.317.1 Detailed Description

[Buffer](#) for [SimulationInputs](#).

6.317.2 Constructor & Destructor Documentation

6.317.2.1 Buffer()

```
Buffer (  
    NetworkProjectConfig cfg )
```

Creates a new [Buffer](#).

Parameters

<i>cfg</i>	Network project configuration.
------------	--------------------------------

6.317.3 Member Function Documentation

6.317.3.1 Add()

```
bool Add (
    SimulationInput input,
    double? insertTime = null )
```

Adds an input to the buffer.

Parameters

<i>input</i>	Input to add.
<i>insertTime</i>	Insert time for <i>input</i> .

Returns

Whether the input was added.

6.317.3.2 Clear()

```
void Clear ( )
```

Clears the buffer.

6.317.3.3 Contains()

```
bool Contains (
    Tick tick )
```

Whether the buffer contains an input for *tick* .

Parameters

<i>tick</i>	Tick to check.
-------------	----------------

Returns

Whether the buffer contains an input for *tick* .

6.317.3.4 CopySortedTo()

```
int CopySortedTo (
    SimulationInput[] array )
```

Copies the buffer to an array and sorts it.

Parameters

<i>array</i>	Array to copy to.
--------------	-------------------

Returns

Number of elements copied.

6.317.3.5 Get()

```
SimulationInput Get (
    Tick tick )
```

Gets the input for *tick* .

Parameters

<i>tick</i>	Tick to get input for.
-------------	------------------------

Returns

Input for *tick* .

6.317.3.6 GetInsertTime()

```
double? GetInsertTime (
    Tick tick )
```

Gets the insert time for *tick* .

Parameters

<i>tick</i>	Tick to get insert time for.
-------------	------------------------------

Returns

Insert time for *tick* .

6.317.3.7 GetLastUsedInputHeader()

```
SimulationInputHeader GetLastUsedInputHeader ( )
```

Retrieves the last used input header data.

Returns

The last used input header data.

6.317.3.8 Remove()

```
bool Remove (
    Tick tick,
    out SimulationInput removed )
```

Removes an input for *tick* from the buffer.

Parameters

<i>tick</i>	Tick to remove.
<i>removed</i>	Removed input.

Returns

Whether an input was removed.

6.317.4 Property Documentation**6.317.4.1 Count**

```
int Count [get]
```

Number of inputs in the buffer.

6.317.4.2 Full

```
bool Full [get]
```

Whether the buffer is full.

6.318 SimulationInputHeader Struct Reference

[Simulation](#) Input Header

Public Attributes

- float [InterpAlpha](#)
Interpolation alpha of the input.
- Tick [InterpFrom](#)
Interpolation from tick of the input.
- Tick [InterpTo](#)
Interpolation to tick of the input.
- Tick [Tick](#)
Tick of the input.

Static Public Attributes

- const int [SIZE](#) = [WORD_COUNT](#) * [Allocator.REPLICATE_WORD_SIZE](#)
Size of the header.
- const int [WORD_COUNT](#) = 4
Word count of the header.

6.318.1 Detailed Description

[Simulation](#) Input Header

6.318.2 Member Data Documentation

6.318.2.1 InterpAlpha

```
float InterpAlpha
```

Interpolation alpha of the input.

6.318.2.2 InterpFrom

`Tick InterpFrom`

Interpolation from tick of the input.

6.318.2.3 InterpTo

`Tick InterpTo`

Interpolation to tick of the input.

6.318.2.4 SIZE

```
const int SIZE = WORD_COUNT * Allocator.REPLICATE_WORD_SIZE [static]
```

Size of the header.

6.318.2.5 Tick

`Tick Tick`

`Tick` of the input.

6.318.2.6 WORD_COUNT

```
const int WORD_COUNT = 4 [static]
```

Word count of the header.

6.319 SimulationMessage Struct Reference

`Simulation` Message

Inherits `ILogDumpable`.

Public Member Functions

- void `ILogDumpable.Dump` (StringBuilder builder)
- bool `GetFlag` (int flag)
 - Get if a flag is set on this [SimulationMessage](#)*
- bool `IsTargeted` ()
 - Signal if this [SimulationMessage](#) is Targeted*
- void `ReferenceCountAdd` ()
 - Add a reference to this [SimulationMessage](#)*
- bool `ReferenceCountSub` ()
 - Subtract a reference from this [SimulationMessage](#)*
- void `SetDummy` ()
 - Set this [SimulationMessage](#) as Dummy*
- void `SetNotTickAligned` ()
 - Set this [SimulationMessage](#) as Not Tick Aligned*
- void `SetStatic` ()
 - Set this [SimulationMessage](#) as Static*
- void `SetTarget` (PlayerRef target)
 - Set the player target of this [SimulationMessage](#)*
- void `SetUnreliable` ()
 - Set this [SimulationMessage](#) as Unreliable*
- override string `ToString` ()
 - [Simulation Message ToString](#)*
- string `ToString` (bool useBrackets)
 - [Simulation Message ToString](#)*

Static Public Member Functions

- static [SimulationMessage](#) * `Allocate` (Simulation sim, int capacityInBytes)
 - Allocate a new [SimulationMessage](#)*
- static bool `CanAllocateUserPayload` (int capacityInBytes)
 - Checks if a message with given size can be allocated.*
- static [SimulationMessage](#) * `Clone` (Simulation sim, [SimulationMessage](#) *message)
 - Create a copy of a [SimulationMessage](#)*
- static byte * `GetData` ([SimulationMessage](#) *message)
 - Get the byte pointer content of a [SimulationMessage](#)*
- static int `ReadInt` ([SimulationMessage](#) *message)
 - Read a int from a [SimulationMessage](#)*
- static [NetworkId](#) `ReadNetworkedObjectRef` ([SimulationMessage](#) *message)
 - Read a [NetworkId](#) from a [SimulationMessage](#)*
- static [Vector3](#) `ReadVector3` ([SimulationMessage](#) *message)
 - Read a [Vector3](#) from a [SimulationMessage](#)*
- static void `WriteInt` ([SimulationMessage](#) *message, int value)
 - Write a int to a [SimulationMessage](#)*
- static void `WriteNetworkedObjectRef` ([SimulationMessage](#) *message, [NetworkId](#) value)
 - Write a [NetworkId](#) to a [SimulationMessage](#)*
- static void `WriteVector3` ([SimulationMessage](#) *message, [Vector3](#) value)
 - Write a [Vector3](#) to a [SimulationMessage](#)*

Public Attributes

- int [Capacity](#)
Capacity in Bits of this [SimulationMessage](#)
- int [Flags](#)
Flags
- int [Offset](#)
Current offset in Bits
- int [References](#)
Reference Count
- [PlayerRef](#) Source
Source Player of this [SimulationMessage](#)
- [PlayerRef](#) Target
Target Player of this [SimulationMessage](#)
- int [Tick](#)
Tick of this [SimulationMessage](#)

Static Public Attributes

- const int [FLAG_DUMMY](#) = 1 << 8
Flag for dummy messages.
- const int [FLAG_INTERNAL](#) = 1 << 6
Flag for internal messages.
- const int [FLAG_NOT_TICK_ALIGNED](#) = 1 << 7
Flag for messages that are not tick aligned.
- const int [FLAG_REMOTE](#) = 1 << 1
Flag for remote messages.
- const int [FLAG_STATIC](#) = 1 << 2
Flag for static messages.
- const int [FLAG_TARGET_PLAYER](#) = 1 << 4
Flag for targeted messages to a player.
- const int [FLAG_TARGET_SERVER](#) = 1 << 5
Flag for targeted messages to the server.
- const int [FLAG_UNRELIABLE](#) = 1 << 3
Flag for unreliable messages.
- const int [FLAG_USER_FLAGS_START](#) = 1 << 16
Flag for user flags.
- const int [FLAG_USER_MESSAGE](#) = 1 << 0
Flag for user messages.
- const int [FLAGS_RESERVED](#) = 0xFFFF
Flag for reserved flags.
- const int [FLAGS_RESERVED_BITS](#) = 16
Flag for reserved bits.
- const int [MAX_PAYLOAD_SIZE](#) = 512
Max user message size in bytes.
- const int [SIZE](#) = 28
[SimulationMessage](#) size in bytes.

Properties

- bool `IsUnreliable` [get]
Signal if this `SimulationMessage` is Unreliable

6.319.1 Detailed Description

`Simulation` Message

6.319.2 Member Function Documentation

6.319.2.1 Allocate()

```
static SimulationMessage* Allocate (
    Simulation sim,
    int capacityInBytes ) [static]
```

Allocate a new `SimulationMessage`

Parameters

<code>sim</code>	<code>Simulation</code> to get the Memory from
<code>capacityInBytes</code>	Size in bytes of the new <code>SimulationMessage</code>

Returns

Pointer to the new `SimulationMessage`

6.319.2.2 CanAllocateUserPayload()

```
static bool CanAllocateUserPayload (
    int capacityInBytes ) [static]
```

Checks if a message with given size can be allocated.

6.319.2.3 Clone()

```
static SimulationMessage* Clone (
    Simulation sim,
    SimulationMessage * message ) [static]
```

Create a copy of a `SimulationMessage`

Parameters

<i>sim</i>	Simulation to allocate from
<i>message</i>	SimulationMessage to copy

Returns

Copy of the [SimulationMessage](#)

6.319.2.4 GetData()

```
static byte* GetData (
    SimulationMessage * message ) [static]
```

Get the byte pointer content of a [SimulationMessage](#)

Parameters

<i>message</i>	SimulationMessage to get the byte pointer of
----------------	--

Returns

Byte pointer of the [SimulationMessage](#)

6.319.2.5 GetFlag()

```
bool GetFlag (
    int flag )
```

Get if a flag is set on this [SimulationMessage](#)

Parameters

<i>flag</i>	Flag to check
-------------	---------------

Returns

True if the flag is set

6.319.2.6 IsTargeted()

```
bool IsTargeted ( )
```

Signal if this [SimulationMessage](#) is Targeted

Returns

True if this [SimulationMessage](#) is Targeted

6.319.2.7 ReadInt()

```
static int ReadInt (
    SimulationMessage * message ) [static]
```

Read a int from a [SimulationMessage](#)

Parameters

<i>message</i>	SimulationMessage to read from
----------------	--

Returns

int read

6.319.2.8 ReadNetworkedObjectRef()

```
static NetworkId ReadNetworkedObjectRef (
    SimulationMessage * message ) [static]
```

Read a [NetworkId](#) from a [SimulationMessage](#)

Parameters

<i>message</i>	SimulationMessage to read from
----------------	--

Returns

[NetworkId](#) read

6.319.2.9 ReadVector3()

```
static Vector3 ReadVector3 (
    SimulationMessage * message ) [static]
```

Read a Vector3 from a [SimulationMessage](#)

Parameters

<i>message</i>	SimulationMessage to read from
----------------	--

Returns

Vector3 read

6.319.2.10 ReferenceCountAdd()

```
void ReferenceCountAdd ( )
```

Add a reference to this [SimulationMessage](#)

6.319.2.11 ReferenceCountSub()

```
bool ReferenceCountSub ( )
```

Subtract a reference from this [SimulationMessage](#)

Returns

True if the reference count is now 0

6.319.2.12 SetDummy()

```
void SetDummy ( )
```

Set this [SimulationMessage](#) as Dummy

6.319.2.13 SetNotTickAligned()

```
void SetNotTickAligned ( )
```

Set this [SimulationMessage](#) as Not Tick Aligned

6.319.2.14 SetStatic()

```
void SetStatic ( )
```

Set this [SimulationMessage](#) as Static

6.319.2.15 SetTarget()

```
void SetTarget (
    PlayerRef target )
```

Set the player target of this [SimulationMessage](#)

Parameters

<i>target</i>	Target Player
---------------	---------------

6.319.2.16 SetUnreliable()

```
void SetUnreliable ( )
```

Set this [SimulationMessage](#) as Unreliable

6.319.2.17 ToString() [1/2]

```
override string ToString ( )
```

[Simulation](#) Message ToString

6.319.2.18 ToString() [2/2]

```
string ToString (
    bool useBrackets )
```

[Simulation](#) Message ToString

6.319.2.19 WriteInt()

```
static void WriteInt (
    SimulationMessage * message,
    int value ) [static]
```

Write a int to a [SimulationMessage](#)

Parameters

<i>message</i>	SimulationMessage to write to
<i>value</i>	int to write

6.319.2.20 WriteNetworkedObjectRef()

```
static void WriteNetworkedObjectRef (
    SimulationMessage * message,
    NetworkId value ) [static]
```

Write a [NetworkId](#) to a [SimulationMessage](#)

Parameters

<i>message</i>	SimulationMessage to write to
<i>value</i>	NetworkId to write

6.319.2.21 WriteVector3()

```
static void WriteVector3 (
    SimulationMessage * message,
    Vector3 value ) [static]
```

Write a [Vector3](#) to a [SimulationMessage](#)

Parameters

<i>message</i>	SimulationMessage to write to
<i>value</i>	Vector3 to write

6.319.3 Member Data Documentation

6.319.3.1 Capacity

```
int Capacity
```

Capacity in Bits of this [SimulationMessage](#)

6.319.3.2 FLAG_DUMMY

```
const int FLAG_DUMMY = 1 << 8 [static]
```

Flag for dummy messages.

6.319.3.3 FLAG_INTERNAL

```
const int FLAG_INTERNAL = 1 << 6 [static]
```

Flag for internal messages.

6.319.3.4 FLAG_NOT_TICK_ALIGNED

```
const int FLAG_NOT_TICK_ALIGNED = 1 << 7 [static]
```

Flag for messages that are not tick aligned.

6.319.3.5 FLAG_REMOTE

```
const int FLAG_REMOTE = 1 << 1 [static]
```

Flag for remote messages.

6.319.3.6 FLAG_STATIC

```
const int FLAG_STATIC = 1 << 2 [static]
```

Flag for static messages.

6.319.3.7 FLAG_TARGET_PLAYER

```
const int FLAG_TARGET_PLAYER = 1 << 4 [static]
```

Flag for targeted messages to a player.

6.319.3.8 FLAG_TARGET_SERVER

```
const int FLAG_TARGET_SERVER = 1 << 5 [static]
```

Flag for targeted messages to the server.

6.319.3.9 FLAG_UNRELIABLE

```
const int FLAG_UNRELIABLE = 1 << 3 [static]
```

Flag for unreliable messages.

6.319.3.10 FLAG_USER_FLAGS_START

```
const int FLAG_USER_FLAGS_START = 1 << 16 [static]
```

Flag for user flags.

6.319.3.11 FLAG_USER_MESSAGE

```
const int FLAG_USER_MESSAGE = 1 << 0 [static]
```

Flag for user messages.

6.319.3.12 Flags

```
int Flags
```

Flags

6.319.3.13 FLAGS_RESERVED

```
const int FLAGS_RESERVED = 0xFFFF [static]
```

Flag for reserved flags.

6.319.3.14 FLAGS_RESERVED_BITS

```
const int FLAGS_RESERVED_BITS = 16 [static]
```

Flag for reserved bits.

6.319.3.15 MAX_PAYLOAD_SIZE

```
const int MAX_PAYLOAD_SIZE = 512 [static]
```

Max user message size in bytes.

6.319.3.16 Offset

```
int Offset
```

Current offset in Bits

6.319.3.17 References

```
int References
```

Reference Count

6.319.3.18 SIZE

```
const int SIZE = 28 [static]
```

[SimulationMessage](#) size in bytes.

6.319.3.19 Source

```
PlayerRef Source
```

Source Player of this [SimulationMessage](#)

6.319.3.20 Target

```
PlayerRef Target
```

Target Player of this [SimulationMessage](#)

6.319.3.21 Tick

int Tick

Tick of this [SimulationMessage](#)

6.319.4 Property Documentation

6.319.4.1 IsUnreliable

bool IsUnreliable [get]

Signal if this [SimulationMessage](#) is Unreliable

6.320 SimulationMessagePtr Struct Reference

[Simulation](#) Message Pointer

Public Attributes

- [SimulationMessage](#) * Message
Pointer to the message.

6.320.1 Detailed Description

[Simulation](#) Message Pointer

6.320.2 Member Data Documentation

6.320.2.1 Message

[SimulationMessage](#)* Message

Pointer to the message.

6.321 SimulationRuntimeConfig Struct Reference

Stores the runtime configuration of the simulation

Public Attributes

- [PlayerRef HostPlayer](#)
Current master client (in shared mode)
- [PlayerRef MasterClient](#)
Current master client (in shared mode)
- `int` [PlayerMaxCount](#)
Current player count
- [SimulationModes ServerMode](#)
Current Simulation Mode
- [TickRate.Resolved TickRate](#)
Current tick rates and send rates for server and client
- [Topologies Topology](#)
Current master client (in shared mode)

6.321.1 Detailed Description

Stores the runtime configuration of the simulation

6.321.2 Member Data Documentation

6.321.2.1 HostPlayer

[PlayerRef](#) HostPlayer

Current master client (in shared mode)

6.321.2.2 MasterClient

[PlayerRef](#) MasterClient

Current master client (in shared mode)

6.321.2.3 PlayerMaxCount

`int` PlayerMaxCount

Current player count

6.321.2.4 ServerMode

`SimulationModes` `ServerMode`

Current `Simulation` Mode

6.321.2.5 TickRate

`TickRate.Resolved` `TickRate`

Current tick rates and send rates for server and client

6.321.2.6 Topology

`Topologies` `Topology`

Current master client (in shared mode)

6.322 INetBitWriteStream Interface Reference

Interface for writing bits to a stream.

Inherited by `NetBitBuffer`, and `NetBitBufferNull`.

Public Member Functions

- bool `WriteBoolean` (bool b)
Writes a boolean value to the stream.
- void `WriteBytesAligned` (void *buffer, int length)
Writes a sequence of bytes to the stream, aligned to byte boundaries.
- void `WriteInt32` (int value, int bits=32)
Writes a 32-bit signed integer to the stream.
- void `WriteInt32VarLength` (int value)
Writes a 32-bit signed integer to the stream with variable length encoding.
- void `WriteInt32VarLength` (int value, int blockSize)
Writes a 32-bit signed integer to the stream with variable length encoding and a specified block size.
- void `WriteUInt64VarLength` (ulong value, int blockSize)
Writes a 64-bit unsigned integer to the stream with variable length encoding and a specified block size.

Properties

- int `OffsetBits` [get]
Gets the current offset in bits.

6.322.1 Detailed Description

Interface for writing bits to a stream.

6.322.2 Member Function Documentation

6.322.2.1 WriteBoolean()

```
bool WriteBoolean (  
    bool b )
```

Writes a boolean value to the stream.

Parameters

<i>b</i>	The boolean value to write.
----------	-----------------------------

Returns

The boolean value that was written.

Implemented in [NetBitBuffer](#), and [NetBitBufferNull](#).

6.322.2.2 WriteBytesAligned()

```
void WriteBytesAligned (  
    void * buffer,  
    int length )
```

Writes a sequence of bytes to the stream, aligned to byte boundaries.

Parameters

<i>buffer</i>	A pointer to the buffer containing the bytes to write.
<i>length</i>	The number of bytes to write.

Implemented in [NetBitBufferNull](#), and [NetBitBuffer](#).

6.322.2.3 WriteInt32()

```
void WriteInt32 (  
    int value,  
    int bits = 32 )
```

Writes a 32-bit signed integer to the stream.

Parameters

<i>value</i>	The 32-bit signed integer value to write.
<i>bits</i>	The number of bits to write. Default is 32.

Implemented in [NetBitBufferNull](#), and [NetBitBuffer](#).

6.322.2.4 WriteInt32VarLength() [1/2]

```
void WriteInt32VarLength (
    int value )
```

Writes a 32-bit signed integer to the stream with variable length encoding.

Parameters

<i>value</i>	The 32-bit signed integer value to write.
--------------	---

Implemented in [NetBitBufferNull](#), and [NetBitBuffer](#).

6.322.2.5 WriteInt32VarLength() [2/2]

```
void WriteInt32VarLength (
    int value,
    int blockSize )
```

Writes a 32-bit signed integer to the stream with variable length encoding and a specified block size.

Parameters

<i>value</i>	The 32-bit signed integer value to write.
<i>blockSize</i>	The block size for variable length encoding.

Implemented in [NetBitBufferNull](#), and [NetBitBuffer](#).

6.322.2.6 WriteUInt64VarLength()

```
void WriteUInt64VarLength (
    ulong value,
    int blockSize )
```

Writes a 64-bit unsigned integer to the stream with variable length encoding and a specified block size.

Parameters

<i>value</i>	The 64-bit unsigned integer value to write.
<i>blockSize</i>	The block size for variable length encoding.

Implemented in [NetBitBufferNull](#), and [NetBitBuffer](#).

6.322.3 Property Documentation

6.322.3.1 OffsetBits

```
int OffsetBits [get]
```

Gets the current offset in bits.

6.323 INetPeerGroupCallbacks Interface Reference

Defines the callbacks for network peer group events.

Inherited by [Simulation](#).

Public Member Functions

- void [OnConnected](#) ([NetConnection](#) *connection)
Called when a connection is established.
- void [OnConnectionAttempt](#) ([NetConnection](#) *connection, int attempts, int totalConnectAttempts)
Called when a connection attempt is made.
- void [OnConnectionFailed](#) ([NetAddress](#) address, [NetConnectFailedReason](#) reason)
Called when a connection attempt fails.
- [OnConnectionRequestReply](#) [OnConnectionRequest](#) ([NetAddress](#) remoteAddress, byte[] token, byte[] uniqueld)
Called when a connection request is received.
- void [OnDisconnected](#) ([NetConnection](#) *connection, [NetDisconnectReason](#) reason)
Called when a connection is disconnected.
- void [OnNotifyData](#) ([NetConnection](#) *connection, [NetBitBuffer](#) *buffer)
Called when notify data is received.
- void [OnNotifyDelivered](#) ([NetConnection](#) *connection, [NetSendEnvelope](#) envelope)
Called when notify data is delivered.
- void [OnNotifyDispose](#) ([NetSendEnvelope](#) envelope)
Called when a notify envelope is disposed.
- void [OnNotifyLost](#) ([NetConnection](#) *connection, [NetSendEnvelope](#) envelope)
Called when notify data is lost.
- void [OnReliableData](#) ([NetConnection](#) *connection, [ReliableId](#) id, byte *data)
Called when reliable data is received.
- void [OnUnconnectedData](#) ([NetBitBuffer](#) *buffer)
Called when unconnected data is received.
- void [OnUnreliableData](#) ([NetConnection](#) *connection, [NetBitBuffer](#) *buffer)
Called when unreliable data is received.

6.323.1 Detailed Description

Defines the callbacks for network peer group events.

6.323.2 Member Function Documentation

6.323.2.1 OnConnected()

```
void OnConnected (
    NetConnection * connection )
```

Called when a connection is established.

Parameters

<i>connection</i>	The connection that was established.
-------------------	--------------------------------------

6.323.2.2 OnConnectionAttempt()

```
void OnConnectionAttempt (
    NetConnection * connection,
    int attempts,
    int totalConnectAttempts )
```

Called when a connection attempt is made.

Parameters

<i>connection</i>	The connection being attempted.
<i>attempts</i>	The number of attempts made.
<i>totalConnectAttempts</i>	The total number of connection attempts.

6.323.2.3 OnConnectionFailed()

```
void OnConnectionFailed (
    NetAddress address,
    NetConnectFailedReason reason )
```

Called when a connection attempt fails.

Parameters

<i>address</i>	The address of the remote peer.
<i>reason</i>	The reason for the connection failure.

6.323.2.4 OnConnectionRequest()

```
OnConnectionRequestReply OnConnectionRequest (
    NetAddress remoteAddress,
    byte[] token,
    byte[] uniqueId )
```

Called when a connection request is received.

Parameters

<i>remoteAddress</i>	The address of the remote peer.
<i>token</i>	The token associated with the connection request.
<i>uniqueId</i>	The unique identifier for the connection request.

Returns

The reply to the connection request.

6.323.2.5 OnDisconnected()

```
void OnDisconnected (
    NetConnection * connection,
    NetDisconnectReason reason )
```

Called when a connection is disconnected.

Parameters

<i>connection</i>	The connection that was disconnected.
<i>reason</i>	The reason for disconnection.

6.323.2.6 OnNotifyData()

```
void OnNotifyData (
    NetConnection * connection,
    NetBitBuffer * buffer )
```

Called when notify data is received.

Parameters

<i>connection</i>	The connection from which the data was received.
<i>buffer</i>	The buffer containing the data.

6.323.2.7 OnNotifyDelivered()

```
void OnNotifyDelivered (
    NetConnection * connection,
    NetSendEnvelope envelope )
```

Called when notify data is delivered.

Parameters

<i>connection</i>	The connection to which the data was delivered.
<i>envelope</i>	The envelope containing the delivered data.

6.323.2.8 OnNotifyDispose()

```
void OnNotifyDispose (
    NetSendEnvelope envelope )
```

Called when a notify envelope is disposed.

Parameters

<i>envelope</i>	The envelope that was disposed.
-----------------	---------------------------------

6.323.2.9 OnNotifyLost()

```
void OnNotifyLost (
    NetConnection * connection,
    NetSendEnvelope envelope )
```

Called when notify data is lost.

Parameters

<i>connection</i>	The connection from which the data was lost.
<i>envelope</i>	The envelope containing the lost data.

6.323.2.10 OnReliableData()

```
void OnReliableData (
    NetConnection * connection,
    ReliableId id,
    byte * data )
```

Called when reliable data is received.

Parameters

<i>connection</i>	The connection from which the data was received.
<i>id</i>	The identifier of the reliable data.
<i>data</i>	The data received.

6.323.2.11 OnUnconnectedData()

```
void OnUnconnectedData (
    NetBitBuffer * buffer )
```

Called when unconnected data is received.

Parameters

<i>buffer</i>	The buffer containing the data.
---------------	---------------------------------

6.323.2.12 OnUnreliableData()

```
void OnUnreliableData (
    NetConnection * connection,
    NetBitBuffer * buffer )
```

Called when unreliable data is received.

Parameters

<i>connection</i>	The connection from which the data was received.
<i>buffer</i>	The buffer containing the data.

6.324 INetSocket Interface Reference

Defines the interface for network socket operations.

Public Member Functions

- [NetAddress Bind](#) ([NetSocket](#) socket, [NetConfig](#) config)
Binds the socket to the specified address and configuration.
- bool [CanFragment](#) ([NetAddress](#) address)
Determines whether the specified address can be fragmented.
- [NetSocket Create](#) ([NetConfig](#) config)
Creates a new socket with the specified configuration.
- void [DeleteEncryptionKey](#) ([NetAddress](#) address)
Deletes the encryption key for the specified address.
- void [Destroy](#) ([NetSocket](#) socket)
Destroys the specified socket.
- void [Initialize](#) ([NetConfig](#) config)
Initializes the socket with the specified configuration.
- bool [Poll](#) ([NetSocket](#) socket, long timeout)
Polls the socket for incoming data with the specified timeout.
- int [Receive](#) ([NetSocket](#) socket, [NetAddress](#) *address, byte *buffer, int bufferLength)
Receives data from the socket into the specified buffer.
- int [Send](#) ([NetSocket](#) socket, [NetAddress](#) *address, byte *buffer, int bufferLength, bool reliable=false)
Sends data from the specified buffer to the socket.
- void [SetupEncryption](#) (byte[] key, byte[] encryptedKey)
Sets up encryption with the specified key and encrypted key.

Properties

- bool [SupportsMultiThreading](#) [get]
Gets a value indicating whether the socket supports multi-threading.

6.324.1 Detailed Description

Defines the interface for network socket operations.

6.324.2 Member Function Documentation

6.324.2.1 Bind()

```
NetAddress Bind (  
    NetSocket socket,  
    NetConfig config )
```

Binds the socket to the specified address and configuration.

Parameters

<i>socket</i>	The socket to bind.
<i>config</i>	The configuration for the socket.

Returns

The bound address.

6.324.2.2 CanFragment()

```
bool CanFragment (
    NetAddress address )
```

Determines whether the specified address can be fragmented.

Parameters

<i>address</i>	The address to check.
----------------	-----------------------

Returns

True if the address can be fragmented, otherwise false.

6.324.2.3 Create()

```
NetSocket Create (
    NetConfig config )
```

Creates a new socket with the specified configuration.

Parameters

<i>config</i>	The configuration for the socket.
---------------	-----------------------------------

Returns

The created socket.

6.324.2.4 DeleteEncryptionKey()

```
void DeleteEncryptionKey (
    NetAddress address )
```

Deletes the encryption key for the specified address.

Parameters

<i>address</i>	The address to delete the encryption key for.
----------------	---

6.324.2.5 Destroy()

```
void Destroy (
    NetSocket socket )
```

Destroys the specified socket.

Parameters

<i>socket</i>	The socket to destroy.
---------------	------------------------

6.324.2.6 Initialize()

```
void Initialize (
    NetConfig config )
```

Initializes the socket with the specified configuration.

Parameters

<i>config</i>	The configuration for the socket.
---------------	-----------------------------------

6.324.2.7 Poll()

```
bool Poll (
    NetSocket socket,
    long timeout )
```

Polls the socket for incoming data with the specified timeout.

Parameters

<i>socket</i>	The socket to poll.
<i>timeout</i>	The timeout in milliseconds.

Returns

True if data is available, otherwise false.

6.324.2.8 Receive()

```
int Receive (  
    NetSocket socket,  
    NetAddress * address,  
    byte * buffer,  
    int bufferLength )
```

Receives data from the socket into the specified buffer.

Parameters

<i>socket</i>	The socket to receive data from.
<i>address</i>	The address of the sender.
<i>buffer</i>	The buffer to store the received data.
<i>bufferLength</i>	The length of the buffer.

Returns

The number of bytes received.

6.324.2.9 Send()

```
int Send (  
    NetSocket socket,  
    NetAddress * address,  
    byte * buffer,  
    int bufferLength,  
    bool reliable = false )
```

Sends data from the specified buffer to the socket.

Parameters

<i>socket</i>	The socket to send data to.
<i>address</i>	The address of the recipient.
<i>buffer</i>	The buffer containing the data to send.
<i>bufferLength</i>	The length of the buffer.
<i>reliable</i>	Send reliable or not

Returns

The number of bytes sent.

6.324.2.10 SetupEncryption()

```
void SetupEncryption (
    byte[] key,
    byte[] encryptedKey )
```

Sets up encryption with the specified key and encrypted key.

Parameters

<i>key</i>	The encryption key.
<i>encryptedKey</i>	The encrypted key.

6.324.3 Property Documentation**6.324.3.1 SupportsMultiThreading**

```
bool SupportsMultiThreading [get]
```

Gets a value indicating whether the socket supports multi-threading.

6.325 NetAddress Struct Reference

Represents a Network Address, which includes a IP and Port This can contains either a IPv4 or a IPv6 address

Inherits `IEquatable< NetAddress >`.

Classes

- class [EqualityComparer](#)
Provides methods to compare two [NetAddress](#) instances for equality.

Public Member Functions

- bool [Equals](#) ([NetAddress](#) other)
Determines whether the specified [NetAddress](#) instances are equal.
- override bool [Equals](#) (object obj)
Determines whether the specified [NetAddress](#) is equal to the current [NetAddress](#).
- override int [GetHashCode](#) ()
Returns a hash code for this instance.
- override string [ToString](#) ()
Provides a string representation of the [NetAddress](#)

Static Public Member Functions

- static [NetAddress Any](#) (ushort port=0)
Create a new [NetAddress](#) using the "Any" IPv4 Address representation (0.0.0.0) with the Port passed as argument
- static [NetAddress AnyIPv6](#) (ushort port=0)
Create a new [NetAddress](#) using the "Any" IPv6 Address representation (::) with the Port passed as argument
- static [NetAddress CreateFromIpPort](#) (string ip, ushort port)
Create a new [NetAddress](#) based on the IP and Port passed as argument
- static [NetAddress FromActorId](#) (int actorId)
Build a new [NetAddress](#) based on an ActorId
- static [NetAddress LocalhostIPv4](#) (ushort port=0)
Create a new [NetAddress](#) on the LocalHost address with the desired Port
- static [NetAddress LocalhostIPv6](#) (ushort port=0)
Create a new [NetAddress](#) on the LocalHost IPv6 Address with the desired Port

Public Attributes

- int [_actorId](#)

Properties

- int [ActorId](#) [get]
Retrieves the Remote Actor ID which this [NetAddress](#) Represents
- bool [HasAddress](#) [get]
Signal if this [NetAddress](#) has a valid IP Address
- bool [IsIPv4](#) [get]
Signal if the [NetAddress](#) represents an IPv4 Address
- bool [IsIPv6](#) [get]
Signal if the [NetAddress](#) represents an IPv6 Address
- bool [IsRelayAddr](#) [get]
Signal if the [NetAddress](#) is a Relayed connection
- bool [IsValid](#) [get]
Signal if this [NetAddress](#) is not default/empty

6.325.1 Detailed Description

Represents a Network Address, which includes a IP and Port This can contains either a IPv4 or a IPv6 address

6.325.2 Member Function Documentation

6.325.2.1 Any()

```
static NetAddress Any (
    ushort port = 0 ) [static]
```

Create a new [NetAddress](#) using the "Any" IPv4 Address representation (0.0.0.0) with the Port passed as argument

Parameters

<i>port</i>	Port used to build the NetAddress
-------------	---

Returns

New [NetAddress](#) reference

6.325.2.2 AnyIPv6()

```
static NetAddress AnyIPv6 (
    ushort port = 0 ) [static]
```

Create a new [NetAddress](#) using the "Any" IPv6 Address representation (::) with the Port passed as argument

Parameters

<i>port</i>	Port used to build the NetAddress
-------------	---

Returns

New [NetAddress](#) reference

6.325.2.3 CreateFromIpPort()

```
static NetAddress CreateFromIpPort (
    string ip,
    ushort port ) [static]
```

Create a new [NetAddress](#) based on the IP and Port passed as argument

Parameters

<i>ip</i>	String representation of an IP, either IPv4 or IPv6
<i>port</i>	Port used to build the NetAddress

Returns

New [NetAddress](#) reference

Exceptions

<i>ArgumentException</i>	If IP is empty/null or an invalid IP, or port < 0
<i>AssertException</i>	If unable to parse IP

6.325.2.4 Equals() [1/2]

```
bool Equals (
    NetAddress other )
```

Determines whether the specified [NetAddress](#) instances are equal.

6.325.2.5 Equals() [2/2]

```
override bool Equals (
    object obj )
```

Determines whether the specified [NetAddress](#) is equal to the current [NetAddress](#).

6.325.2.6 FromActorId()

```
static NetAddress FromActorId (
    int actorId ) [static]
```

Build a new [NetAddress](#) based on an ActorId

Parameters

<i>actorId</i>	ActorId used to build the NetAddress
----------------	--

Returns

Relay [NetAddress](#) that references the ActorId

ActorId must be 0 or greater

6.325.2.7 GetHashCode()

```
override int GetHashCode ( )
```

Returns a hash code for this instance.

6.325.2.8 LocalhostIPv4()

```
static NetAddress LocalhostIPv4 (
    ushort port = 0 ) [static]
```

Create a new [NetAddress](#) on the LocalHost address with the desired Port

Parameters

<i>port</i>	Port used to build the NetAddress
-------------	---

Returns

New [NetAddress](#) reference

6.325.2.9 LocalhostIPv6()

```
static NetAddress LocalhostIPv6 (
    ushort port = 0 ) [static]
```

Create a new [NetAddress](#) on the LocalHost IPv6 Address with the desired Port

Parameters

<i>port</i>	Port used to build the NetAddress
-------------	---

Returns

New [NetAddress](#) reference

6.325.2.10 ToString()

```
override string ToString ( )
```

Provides a string representation of the [NetAddress](#)

6.325.3 Property Documentation**6.325.3.1 ActorId**

```
int ActorId [get]
```

Retrieves the Remote Actor ID which this [NetAddress](#) Represents

6.325.3.2 HasAddress

```
bool HasAddress [get]
```

Signal if this [NetAddress](#) has a valid IP Address

6.325.3.3 IsIPv4

```
bool IsIPv4 [get]
```

Signal if the [NetAddress](#) represents an IPv4 Address

6.325.3.4 IsIPv6

```
bool IsIPv6 [get]
```

Signal if the [NetAddress](#) represents an IPv6 Address

6.325.3.5 IsRelayAddr

```
bool IsRelayAddr [get]
```

Signal if the [NetAddress](#) is a Relayed connection

6.325.3.6 IsValid

```
bool IsValid [get]
```

Signal if this [NetAddress](#) is not default/empty

6.326 NetAddress.EqualityComparer Class Reference

Provides methods to compare two [NetAddress](#) instances for equality.

Inherits [IEqualityComparer< NetAddress >](#).

Public Member Functions

- bool [Equals](#) ([NetAddress](#) x, [NetAddress](#) y)
Determines whether the specified [NetAddress](#) instances are equal.
- int [GetHashCode](#) ([NetAddress](#) obj)
Returns a hash code for the specified [NetAddress](#).

6.326.1 Detailed Description

Provides methods to compare two [NetAddress](#) instances for equality.

6.326.2 Member Function Documentation

6.326.2.1 Equals()

```
bool Equals (  
    NetAddress x,  
    NetAddress y )
```

Determines whether the specified [NetAddress](#) instances are equal.

6.326.2.2 GetHashCode()

```
int GetHashCode (  
    NetAddress obj )
```

Returns a hash code for the specified [NetAddress](#).

6.327 NetBitBuffer Struct Reference

Represents a buffer for reading and writing bits.

Inherits [INetBitWriteStream](#), and [ILogDumpable](#).

Classes

- struct [Offset](#)
Represents an offset within a [NetBitBuffer](#).

Public Member Functions

- bool [CanRead](#) (int bits)
Checks if the specified number of bits can be read from the buffer.
- bool [CanWrite](#) (int bits)
Checks if the specified number of bits can be written to the buffer.
- bool [CheckBitCount](#) (int count)
Checks if the specified number of bits can be read or written without exceeding the buffer length.
- void [Clear](#) ()
Clears the buffer by setting all bits to zero.
- void [ILogDumpable.Dump](#) (StringBuilder builder)
- byte * [GetDataPointer](#) ()
Gets a pointer to the current data position in the buffer.
- void [PadToByteBoundary](#) ()
Pads the buffer to the next byte boundary by writing zero bits if necessary.
- byte * [PadToByteBoundaryAndGetPtr](#) ()
Pads the buffer to the next byte boundary and returns a pointer to the current data position.
- ulong [Peek](#) (int bits)
- bool [PeekBoolean](#) ()
Peeks at the next boolean value in the buffer without advancing the read position.
- ulong [Read](#) (int bits)
- bool [ReadBoolean](#) ()
Reads a boolean value from the buffer.
- byte [ReadByte](#) (int bits=8)
Reads a byte value from the buffer.
- void [ReadBytesAligned](#) (byte[] buffer, int length)
Reads an array of bytes from the buffer, ensuring byte alignment.
- void [ReadBytesAligned](#) (void *buffer, int length)
Reads a block of bytes from the buffer, ensuring byte alignment.
- double [ReadDouble](#) ()
Reads a double-precision floating-point value from the buffer.
- short [ReadInt16](#) (int bits=16)
Reads a 16-bit signed integer value from the buffer.
- int [ReadInt32](#) (int bits=32)
Reads a 32-bit signed integer value from the buffer.
- int [ReadInt32VarLength](#) ()
Reads a 32-bit signed integer value with variable length from the buffer.
- int [ReadInt32VarLength](#) (int blockSize)
Reads a 32-bit signed integer value with variable length from the buffer.
- long [ReadInt64](#) (int bits=64)
Reads a 64-bit signed integer value from the buffer.
- long [ReadInt64VarLength](#) (int blockSize)
Reads a 64-bit signed integer value with variable length from the buffer.
- float [ReadSingle](#) ()
Reads a single-precision floating-point value from the buffer.
- string [ReadString](#) ()
Reads a string from the buffer using UTF-8 encoding.
- string [ReadString](#) (Encoding encoding)
Reads a string from the buffer using the specified encoding.
- ushort [ReadUInt16](#) (int bits=16)
Reads a 16-bit unsigned integer value from the buffer.

- uint [ReadUInt32](#) (int bits=32)
Reads a 32-bit unsigned integer value from the buffer.
- uint [ReadUInt32VarLength](#) ()
Reads a 32-bit unsigned integer value with variable length from the buffer.
- uint [ReadUInt32VarLength](#) (int blockSize)
Reads a 32-bit unsigned integer value with variable length from the buffer.
- ulong [ReadUInt64](#) (int bits=64)
Reads a 64-bit unsigned integer value from the buffer.
- ulong [ReadUInt64VarLength](#) (int blockSize)
Reads a 64-bit unsigned integer value with variable length from the buffer.
- void [ReplaceDataFromBlockWithTemp](#) (int tempSize)
Replaces the current data block with a temporary block of the specified size.
- void [SeekToByteBoundary](#) ()
Advances the buffer offset to the next byte boundary.
- void [SetBufferLengthBytes](#) (ulong *buffer, int lengthInBytes)
Sets the buffer length in bytes and updates the internal length in bits.
- void [Write](#) (ulong value, int bits)
Writes a value to the buffer with a specified number of bits.
- bool [WriteBoolean](#) (bool value)
Writes a boolean value to the buffer.
- void [WriteByte](#) (byte value, int bits=8)
Writes a byte value to the buffer.
- void [WriteBytesAligned](#) (byte[] buffer, int length)
Writes an array of bytes to the buffer, ensuring byte alignment.
- void [WriteBytesAligned](#) (void *buffer, int length)
Writes a block of bytes to the buffer, ensuring byte alignment.
- void [WriteDouble](#) (double value)
Writes a double-precision floating-point value to the buffer.
- void [WriteInt16](#) (short value, int bits=16)
Writes a 16-bit signed integer value to the buffer.
- void [WriteInt32](#) (int value, int bits=32)
Writes a 32-bit signed integer value to the buffer.
- void [WriteInt32AtOffset](#) (int value, int offset, int bits)
Writes a 32-bit integer value at a specified offset in the buffer.
- void [WriteInt32VarLength](#) (int value)
Writes a 32-bit signed integer value with variable length to the buffer.
- void [WriteInt32VarLength](#) (int value, int blockSize)
Writes a 32-bit signed integer value with variable length to the buffer.
- void [WriteInt64](#) (long value, int bits=64)
Writes a 64-bit signed integer value to the buffer.
- void [WriteInt64VarLength](#) (long value, int blockSize)
Writes a 64-bit signed integer value with variable length to the buffer.
- void [WriteSingle](#) (float value)
Writes a single-precision floating-point value to the buffer.
- void [WriteSlow](#) (ulong value, int bits)
Writes a value to the buffer with a specified number of bits, handling cases where the value spans multiple words.
- void [WriteString](#) (string value)
Writes a string to the buffer using UTF-8 encoding.
- void [WriteString](#) (string value, Encoding encoding)
Writes a string to the buffer using the specified encoding.
- void [WriteUInt16](#) (ushort value, int bits=16)

- Writes a 16-bit unsigned integer value to the buffer.*

 - void [WriteUInt32](#) (uint value, int bits=32)

Writes a 32-bit unsigned integer value to the buffer.

 - void [WriteUInt32VarLength](#) (uint value)

Writes a 32-bit unsigned integer value with variable length to the buffer.

 - void [WriteUInt32VarLength](#) (uint value, int blockSize)

Writes a 32-bit unsigned integer value with variable length to the buffer.

 - void [WriteUInt64](#) (ulong value, int bits=64)

Writes a 64-bit unsigned integer value to the buffer.

 - void [WriteUInt64AtOffset](#) (ulong value, int offset, int bits)

Writes a 64-bit unsigned integer value at a specified offset in the buffer.

 - void [WriteUInt64VarLength](#) (ulong value, int blockSize)

Writes a 64-bit unsigned integer value with variable length to the buffer.

Static Public Member Functions

- static [NetBitBuffer](#) * [Allocate](#) (int group, int size)

Allocates a new [NetBitBuffer](#) with the specified group and size.
- static [Offset](#) [GetOffset](#) ([NetBitBuffer](#) *buffer)

Gets the current offset of the specified buffer.
- static void [Release](#) ([NetBitBuffer](#) *buffer)

Releases the specified buffer.
- static void [ReleaseRef](#) (ref [NetBitBuffer](#) *buffer)

Releases the reference to the specified buffer and sets it to null.

Public Attributes

- [ulong](#) * [_data](#)
- [ulong](#) * [_dataBlockOriginal](#)
- [int](#) [_group](#)
- [int](#) [_lengthBits](#)
- [int](#) [_lengthBytes](#)
- [int](#) [_offsetBits](#)
- [NetAddress](#) [Address](#)

The address of the buffer.

Static Public Attributes

- const [int](#) [BITCOUNT](#) = 64
- const [int](#) [BYTESHIFT](#) = 3
- const [int](#) [INDEXSHIFT](#) = 6
- const [ulong](#) [MAXVALUE](#) = [ulong.MaxValue](#)
- const [int](#) [USEDMASK](#) = [BITCOUNT](#) - 1

Properties

- int [BytesRemaining](#) [get]
Gets the number of bytes remaining in the buffer.
- `ulong *` [Data](#) [get]
Gets or sets a pointer to the data in the buffer.
- bool [Done](#) [get, set]
Gets a value indicating whether the buffer has been fully read.
- bool [DoneOrOverflow](#) [get, set]
Gets a value indicating whether the buffer is done or has overflowed.
- bool [IsOnEvenByte](#) [get, set]
Gets a value indicating whether the buffer is aligned to an even byte boundary.
- int [LengthBits](#) [get]
Gets the length of the buffer in bits.
- int [LengthBytes](#) [get]
Gets or sets the length of the buffer in bytes.
- bool [MoreToRead](#) [get, set]
Gets a value indicating whether there is more data to read in the buffer.
- int [OffsetBits](#) [get, set]
Gets or sets the current offset in bits.
- int [OffsetBitsUnsafe](#) [get, set]
Gets or sets the current offset in bits without any safety checks.
- int [OffsetBytes](#) [get]
Gets the current offset in bytes.
- bool [Overflow](#) [get, set]
Gets a value indicating whether the buffer has overflowed.

6.327.1 Detailed Description

Represents a buffer for reading and writing bits.

6.327.2 Member Function Documentation

6.327.2.1 Allocate()

```
static NetBitBuffer* Allocate (
    int group,
    int size ) [static]
```

Allocates a new [NetBitBuffer](#) with the specified group and size.

Parameters

<i>group</i>	The group identifier for the buffer.
<i>size</i>	The size of the buffer in bytes.

Returns

A pointer to the allocated [NetBitBuffer](#).

6.327.2.2 CanRead()

```
bool CanRead (
    int bits )
```

Checks if the specified number of bits can be read from the buffer.

Parameters

<i>bits</i>	The number of bits to check.
-------------	------------------------------

Returns

`true` if the specified number of bits can be read; otherwise, `false`.

6.327.2.3 CanWrite()

```
bool CanWrite (
    int bits )
```

Checks if the specified number of bits can be written to the buffer.

Parameters

<i>bits</i>	The number of bits to check.
-------------	------------------------------

Returns

`true` if the specified number of bits can be written; otherwise, `false`.

6.327.2.4 CheckBitCount()

```
bool CheckBitCount (
    int count )
```

Checks if the specified number of bits can be read or written without exceeding the buffer length.

Parameters

<i>count</i>	The number of bits to check.
--------------	------------------------------

Returns

`true` if the specified number of bits can be read or written; otherwise, `false`.

6.327.2.5 Clear()

```
void Clear ( )
```

Clears the buffer by setting all bits to zero.

6.327.2.6 GetDataPointer()

```
byte* GetDataPointer ( )
```

Gets a pointer to the current data position in the buffer.

Returns

A pointer to the current data position in the buffer.

6.327.2.7 GetOffset()

```
static Offset GetOffset (
    NetBitBuffer * buffer ) [static]
```

Gets the current offset of the specified buffer.

Parameters

<i>buffer</i>	The buffer to get the offset from.
---------------	------------------------------------

Returns

An [Offset](#) struct representing the current offset.

6.327.2.8 PadToByteBoundary()

```
void PadToByteBoundary ( )
```

Pads the buffer to the next byte boundary by writing zero bits if necessary.

6.327.2.9 PadToByteBoundaryAndGetPtr()

```
byte* PadToByteBoundaryAndGetPtr ( )
```

Pads the buffer to the next byte boundary and returns a pointer to the current data position.

Returns

A pointer to the current data position in the buffer after padding to the byte boundary.

6.327.2.10 PeekBoolean()

```
bool PeekBoolean ( )
```

Peeks at the next boolean value in the buffer without advancing the read position.

Returns

The boolean value peeked from the buffer.

6.327.2.11 ReadBoolean()

```
bool ReadBoolean ( )
```

Reads a boolean value from the buffer.

Returns

The boolean value read from the buffer.

6.327.2.12 ReadByte()

```
byte ReadByte (
    int bits = 8 )
```

Reads a byte value from the buffer.

Parameters

<i>bits</i>	The number of bits to read. Default is 8.
-------------	---

Returns

The byte value read from the buffer.

6.327.2.13 ReadBytesAligned() [1/2]

```
void ReadBytesAligned (
    byte[] buffer,
    int length )
```

Reads an array of bytes from the buffer, ensuring byte alignment.

Parameters

<i>buffer</i>	The array to store the read bytes.
<i>length</i>	The number of bytes to read.

6.327.2.14 ReadBytesAligned() [2/2]

```
void ReadBytesAligned (
    void * buffer,
    int length )
```

Reads a block of bytes from the buffer, ensuring byte alignment.

Parameters

<i>buffer</i>	A pointer to the block of bytes to store the read data.
<i>length</i>	The number of bytes to read.

6.327.2.15 ReadDouble()

```
double ReadDouble ( )
```

Reads a double-precision floating-point value from the buffer.

Returns

The double-precision floating-point value read from the buffer.

6.327.2.16 ReadInt16()

```
short ReadInt16 (
    int bits = 16 )
```

Reads a 16-bit signed integer value from the buffer.

Parameters

<i>bits</i>	The number of bits to read. Default is 16.
-------------	--

Returns

The 16-bit signed integer value read from the buffer.

6.327.2.17 ReadInt32()

```
int ReadInt32 (
    int bits = 32 )
```

Reads a 32-bit signed integer value from the buffer.

Parameters

<i>bits</i>	The number of bits to read. Default is 32.
-------------	--

Returns

The 32-bit signed integer value read from the buffer.

6.327.2.18 ReadInt32VarLength() [1/2]

```
int ReadInt32VarLength ( )
```

Reads a 32-bit signed integer value with variable length from the buffer.

Returns

The 32-bit signed integer value read from the buffer.

6.327.2.19 ReadInt32VarLength() [2/2]

```
int ReadInt32VarLength (
    int blockSize )
```

Reads a 32-bit signed integer value with variable length from the buffer.

Parameters

<i>blockSize</i>	The block size in bits.
------------------	-------------------------

Returns

The 32-bit signed integer value read from the buffer.

6.327.2.20 ReadInt64()

```
long ReadInt64 (
    int bits = 64 )
```

Reads a 64-bit signed integer value from the buffer.

Parameters

<i>bits</i>	The number of bits to read. Default is 64.
-------------	--

Returns

The 64-bit signed integer value read from the buffer.

6.327.2.21 ReadInt64VarLength()

```
long ReadInt64VarLength (
    int blockSize )
```

Reads a 64-bit signed integer value with variable length from the buffer.

Parameters

<i>blockSize</i>	The block size in bits.
------------------	-------------------------

Returns

The 64-bit signed integer value read from the buffer.

6.327.2.22 ReadSingle()

```
float ReadSingle ( )
```

Reads a single-precision floating-point value from the buffer.

Returns

The single-precision floating-point value read from the buffer.

6.327.2.23 ReadString() [1/2]

```
string ReadString ( )
```

Reads a string from the buffer using UTF-8 encoding.

Returns

The string read from the buffer.

6.327.2.24 ReadString() [2/2]

```
string ReadString (
    Encoding encoding )
```

Reads a string from the buffer using the specified encoding.

Parameters

<i>encoding</i>	The encoding to use.
-----------------	----------------------

Returns

The string read from the buffer.

6.327.2.25 ReadUInt16()

```
ushort ReadUInt16 (
    int bits = 16 )
```

Reads a 16-bit unsigned integer value from the buffer.

Parameters

<i>bits</i>	The number of bits to read. Default is 16.
-------------	--

Returns

The 16-bit unsigned integer value read from the buffer.

6.327.2.26 ReadUInt32()

```
uint ReadUInt32 (
    int bits = 32 )
```

Reads a 32-bit unsigned integer value from the buffer.

Parameters

<i>bits</i>	The number of bits to read. Default is 32.
-------------	--

Returns

The 32-bit unsigned integer value read from the buffer.

6.327.2.27 ReadUInt32VarLength() [1/2]

```
uint ReadUInt32VarLength ( )
```

Reads a 32-bit unsigned integer value with variable length from the buffer.

Returns

The 32-bit unsigned integer value read from the buffer.

6.327.2.28 ReadUInt32VarLength() [2/2]

```
uint ReadUInt32VarLength (
    int blockSize )
```

Reads a 32-bit unsigned integer value with variable length from the buffer.

Parameters

<i>blockSize</i>	The block size in bits.
------------------	-------------------------

Returns

The 32-bit unsigned integer value read from the buffer.

6.327.2.29 ReadUInt64()

```
ulong ReadUInt64 (
    int bits = 64 )
```

Reads a 64-bit unsigned integer value from the buffer.

Parameters

<i>bits</i>	The number of bits to read. Default is 64.
-------------	--

Returns

The 64-bit unsigned integer value read from the buffer.

6.327.2.30 ReadUInt64VarLength()

```
ulong ReadUInt64VarLength (
    int blockSize )
```

Reads a 64-bit unsigned integer value with variable length from the buffer.

Parameters

<i>blockSize</i>	The block size in bits.
------------------	-------------------------

Returns

The 64-bit unsigned integer value read from the buffer.

6.327.2.31 Release()

```
static void Release (
    NetBitBuffer * buffer ) [static]
```

Releases the specified buffer.

Parameters

<i>buffer</i>	The buffer to release.
---------------	------------------------

6.327.2.32 ReleaseRef()

```
static void ReleaseRef (  
    ref NetBitBuffer * buffer ) [static]
```

Releases the reference to the specified buffer and sets it to null.

Parameters

<i>buffer</i>	A reference to the buffer to release.
---------------	---------------------------------------

6.327.2.33 ReplaceDataFromBlockWithTemp()

```
void ReplaceDataFromBlockWithTemp (  
    int tempSize )
```

Replaces the current data block with a temporary block of the specified size.

Parameters

<i>tempSize</i>	The size of the temporary block in bytes.
-----------------	---

6.327.2.34 SeekToByteBoundary()

```
void SeekToByteBoundary ( )
```

Advances the buffer offset to the next byte boundary.

6.327.2.35 SetBufferLengthBytes()

```
void SetBufferLengthBytes (  
    ulong * buffer,  
    int lengthInBytes )
```

Sets the buffer length in bytes and updates the internal length in bits.

Parameters

<i>buffer</i>	A pointer to the buffer.
<i>lengthInBytes</i>	The length of the buffer in bytes.

6.327.2.36 Write()

```
void Write (
    ulong value,
    int bits )
```

Writes a value to the buffer with a specified number of bits.

Parameters

<i>value</i>	The value to write.
<i>bits</i>	The number of bits to write.

6.327.2.37 WriteBoolean()

```
bool WriteBoolean (
    bool value )
```

Writes a boolean value to the buffer.

Parameters

<i>value</i>	The boolean value to write.
--------------	-----------------------------

Returns

The boolean value that was written.

Implements [INetBitWriteStream](#).

6.327.2.38 WriteByte()

```
void WriteByte (
    byte value,
    int bits = 8 )
```

Writes a byte value to the buffer.

Parameters

<i>value</i>	The byte value to write.
<i>bits</i>	The number of bits to write. Default is 8.

6.327.2.39 WriteBytesAligned() [1/2]

```
void WriteBytesAligned (
    byte[] buffer,
    int length )
```

Writes an array of bytes to the buffer, ensuring byte alignment.

Parameters

<i>buffer</i>	The array of bytes to write.
<i>length</i>	The number of bytes to write.

6.327.2.40 WriteBytesAligned() [2/2]

```
void WriteBytesAligned (
    void * buffer,
    int length )
```

Writes a block of bytes to the buffer, ensuring byte alignment.

Parameters

<i>buffer</i>	A pointer to the block of bytes to write.
<i>length</i>	The number of bytes to write.

Implements [INetBitWriteStream](#).

6.327.2.41 WriteDouble()

```
void WriteDouble (
    double value )
```

Writes a double-precision floating-point value to the buffer.

Parameters

<i>value</i>	The double-precision floating-point value to write.
--------------	---

6.327.2.42 WriteInt16()

```
void WriteInt16 (
    short value,
    int bits = 16 )
```

Writes a 16-bit signed integer value to the buffer.

Parameters

<i>value</i>	The 16-bit signed integer value to write.
<i>bits</i>	The number of bits to write. Default is 16.

6.327.2.43 WriteInt32()

```
void WriteInt32 (
    int value,
    int bits = 32 )
```

Writes a 32-bit signed integer value to the buffer.

Parameters

<i>value</i>	The 32-bit signed integer value to write.
<i>bits</i>	The number of bits to write. Default is 32.

Implements [INetBitWriteStream](#).

6.327.2.44 WriteInt32AtOffset()

```
void WriteInt32AtOffset (
    int value,
    int offset,
    int bits )
```

Writes a 32-bit integer value at a specified offset in the buffer.

Parameters

<i>value</i>	The 32-bit integer value to write.
<i>offset</i>	The offset in bits where the value should be written.
<i>bits</i>	The number of bits to write.

6.327.2.45 WriteInt32VarLength() [1/2]

```
void WriteInt32VarLength (
    int value )
```

Writes a 32-bit signed integer value with variable length to the buffer.

Parameters

<i>value</i>	The 32-bit signed integer value to write.
--------------	---

Implements [INetBitWriteStream](#).

6.327.2.46 WriteInt32VarLength() [2/2]

```
void WriteInt32VarLength (
    int value,
    int blockSize )
```

Writes a 32-bit signed integer value with variable length to the buffer.

Parameters

<i>value</i>	The 32-bit signed integer value to write.
<i>blockSize</i>	The block size in bits.

Implements [INetBitWriteStream](#).

6.327.2.47 WriteInt64()

```
void WriteInt64 (
    long value,
    int bits = 64 )
```

Writes a 64-bit signed integer value to the buffer.

Parameters

<i>value</i>	The 64-bit signed integer value to write.
<i>bits</i>	The number of bits to write. Default is 64.

6.327.2.48 WriteInt64VarLength()

```
void WriteInt64VarLength (
    long value,
    int blockSize )
```

Writes a 64-bit signed integer value with variable length to the buffer.

Parameters

<i>value</i>	The 64-bit signed integer value to write.
<i>blockSize</i>	The block size in bits.

6.327.2.49 WriteSingle()

```
void WriteSingle (
    float value )
```

Writes a single-precision floating-point value to the buffer.

Parameters

<i>value</i>	The single-precision floating-point value to write.
--------------	---

6.327.2.50 WriteSlow()

```
void WriteSlow (
    ulong value,
    int bits )
```

Writes a value to the buffer with a specified number of bits, handling cases where the value spans multiple words.

Parameters

<i>value</i>	The value to write.
<i>bits</i>	The number of bits to write.

6.327.2.51 WriteString() [1/2]

```
void WriteString (
    string value )
```

Writes a string to the buffer using UTF-8 encoding.

Parameters

<i>value</i>	The string to write.
--------------	----------------------

6.327.2.52 WriteString() [2/2]

```
void WriteString (
    string value,
    Encoding encoding )
```

Writes a string to the buffer using the specified encoding.

Parameters

<i>value</i>	The string to write.
<i>encoding</i>	The encoding to use.

6.327.2.53 WriteUInt16()

```
void WriteUInt16 (
    ushort value,
    int bits = 16 )
```

Writes a 16-bit unsigned integer value to the buffer.

Parameters

<i>value</i>	The 16-bit unsigned integer value to write.
<i>bits</i>	The number of bits to write. Default is 16.

6.327.2.54 WriteUInt32()

```
void WriteUInt32 (
```

```
uint value,  
int bits = 32 )
```

Writes a 32-bit unsigned integer value to the buffer.

Parameters

<i>value</i>	The 32-bit unsigned integer value to write.
<i>bits</i>	The number of bits to write. Default is 32.

6.327.2.55 WriteUInt32VarLength() [1/2]

```
void WriteUInt32VarLength (  
    uint value )
```

Writes a 32-bit unsigned integer value with variable length to the buffer.

Parameters

<i>value</i>	The 32-bit unsigned integer value to write.
--------------	---

6.327.2.56 WriteUInt32VarLength() [2/2]

```
void WriteUInt32VarLength (  
    uint value,  
    int blockSize )
```

Writes a 32-bit unsigned integer value with variable length to the buffer.

Parameters

<i>value</i>	The 32-bit unsigned integer value to write.
<i>blockSize</i>	The block size in bits.

6.327.2.57 WriteUInt64()

```
void WriteUInt64 (  
    ulong value,  
    int bits = 64 )
```

Writes a 64-bit unsigned integer value to the buffer.

Parameters

<i>value</i>	The 64-bit unsigned integer value to write.
<i>bits</i>	The number of bits to write. Default is 64.

6.327.2.58 WriteUInt64AtOffset()

```
void WriteUInt64AtOffset (
    ulong value,
    int offset,
    int bits )
```

Writes a 64-bit unsigned integer value at a specified offset in the buffer.

Parameters

<i>value</i>	The 64-bit unsigned integer value to write.
<i>offset</i>	The offset in bits where the value should be written.
<i>bits</i>	The number of bits to write.

6.327.2.59 WriteUInt64VarLength()

```
void WriteUInt64VarLength (
    ulong value,
    int blockSize )
```

Writes a 64-bit unsigned integer value with variable length to the buffer.

Parameters

<i>value</i>	The 64-bit unsigned integer value to write.
<i>blockSize</i>	The block size in bits.

Implements [INetBitWriteStream](#).

6.327.3 Member Data Documentation**6.327.3.1 Address**

[NetAddress](#) Address

The address of the buffer.

6.327.4 Property Documentation

6.327.4.1 BytesRemaining

```
int BytesRemaining [get]
```

Gets the number of bytes remaining in the buffer.

6.327.4.2 Data

```
ulong* Data [get]
```

Gets or sets a pointer to the data in the buffer.

6.327.4.3 Done

```
bool Done [get], [set]
```

Gets a value indicating whether the buffer has been fully read.

6.327.4.4 DoneOrOverflow

```
bool DoneOrOverflow [get], [set]
```

Gets a value indicating whether the buffer is done or has overflowed.

6.327.4.5 IsOnEvenByte

```
bool IsOnEvenByte [get], [set]
```

Gets a value indicating whether the buffer is aligned to an even byte boundary.

6.327.4.6 LengthBits

```
int LengthBits [get]
```

Gets the length of the buffer in bits.

6.327.4.7 LengthBytes

```
int LengthBytes [get]
```

Gets or sets the length of the buffer in bytes.

6.327.4.8 MoreToRead

```
bool MoreToRead [get], [set]
```

Gets a value indicating whether there is more data to read in the buffer.

6.327.4.9 OffsetBits

```
int OffsetBits [get], [set]
```

Gets or sets the current offset in bits.

6.327.4.10 OffsetBitsUnsafe

```
int OffsetBitsUnsafe [get], [set]
```

Gets or sets the current offset in bits without any safety checks.

6.327.4.11 OffsetBytes

```
int OffsetBytes [get]
```

Gets the current offset in bytes.

6.327.4.12 Overflow

```
bool Overflow [get], [set]
```

Gets a value indicating whether the buffer has overflowed.

6.328 NetBitBuffer.Offset Struct Reference

Represents an offset within a [NetBitBuffer](#).

Public Member Functions

- int [GetLength](#) ([NetBitBuffer](#) *buffer)
Gets the length in bits from the stored offset to the current offset of the buffer.
- [Offset](#) ([NetBitBuffer](#) *buffer)
Initializes a new instance of the [Offset](#) struct with the specified buffer.

Public Attributes

- int `_offset`

6.328.1 Detailed Description

Represents an offset within a [NetBitBuffer](#).

6.328.2 Constructor & Destructor Documentation

6.328.2.1 Offset()

```
Offset (  
    NetBitBuffer * buffer )
```

Initializes a new instance of the [Offset](#) struct with the specified buffer.

Parameters

<code>buffer</code>	The buffer to initialize the offset from.
---------------------	---

6.328.3 Member Function Documentation

6.328.3.1 GetLength()

```
int GetLength (
    NetBitBuffer * buffer )
```

Gets the length in bits from the stored offset to the current offset of the buffer.

Parameters

<i>buffer</i>	The buffer to calculate the length from.
---------------	--

Returns

The length in bits.

6.329 NetBitBufferList Struct Reference

Represents a linked list of [Fusion.Sockets.NetBitBuffer](#)

Public Member Functions

- void [AddFirst](#) ([NetBitBuffer](#) *item)
Add a [Fusion.Sockets.NetBitBuffer](#) at the beginning of the List
- void [AddLast](#) ([NetBitBuffer](#) *item)
Add a [Fusion.Sockets.NetBitBuffer](#) at the end of the list.
- bool [IsInList](#) ([NetBitBuffer](#) *item)
Check if a specific [Fusion.Sockets.NetBitBuffer](#) is in the list
- void [Remove](#) ([NetBitBuffer](#) *item)
Remove a specific [Fusion.Sockets.NetBitBuffer](#) from the list
- [NetBitBuffer](#) * [RemoveHead](#) ()
Removes the first element of the list

Public Attributes

- int **Count**
- [NetBitBuffer](#) * **Head**
- [NetBitBuffer](#) * **Tail**

6.329.1 Detailed Description

Represents a linked list of [Fusion.Sockets.NetBitBuffer](#)

6.329.2 Member Function Documentation

6.329.2.1 AddFirst()

```
void AddFirst (
    NetBitBuffer * item )
```

Add a [Fusion.Sockets.NetBitBuffer](#) at the beginning of the List

Parameters

<i>item</i>	NetBitBuffer to add to the list
-------------	---

6.329.2.2 AddLast()

```
void AddLast (  
    NetBitBuffer * item )
```

Add a [Fusion.Sockets.NetBitBuffer](#) at the end of the list.

Parameters

<i>item</i>	NetBitBuffer to add to the list
-------------	---

6.329.2.3 IsInList()

```
bool IsInList (  
    NetBitBuffer * item )
```

Check if a specific [Fusion.Sockets.NetBitBuffer](#) is in the list

Parameters

<i>item</i>	NetBitBuffer to check
-------------	---------------------------------------

Returns

True if the list contains the item, false otherwise

6.329.2.4 Remove()

```
void Remove (  
    NetBitBuffer * item )
```

Remove a specific [Fusion.Sockets.NetBitBuffer](#) from the list

Parameters

<i>item</i>	NetBitBuffer to remove
-------------	--

6.329.2.5 RemoveHead()

`NetBitBuffer*` RemoveHead ()

Removes the first element of the list

Returns

`NetBitBuffer` reference

6.330 NetBitBufferNull Struct Reference

Represents a null bit buffer for writing data.

Inherits [INetBitWriteStream](#).

Public Member Functions

- void [PadToByteBoundary](#) ()
Pads the buffer to the next byte boundary by writing zero bits if necessary.
- bool [WriteBoolean](#) (bool b)
Writes a boolean value to the buffer and increments the offset by 1 bit.
- void [WriteByte](#) (byte value, int bits=8)
Writes a byte value to the buffer and increments the offset by the specified number of bits.
- unsafe void [WriteBytesAligned](#) (void *buffer, int length)
Writes a specified number of bytes from a buffer to the bit buffer, ensuring byte alignment.
- void [WriteInt32](#) (int value, int bits=32)
Writes a 32-bit integer value to the buffer and increments the offset by the specified number of bits.
- void [WriteInt32VarLength](#) (int value)
Writes a 32-bit integer value with variable length encoding.
- void [WriteInt32VarLength](#) (int value, int blockSize)
Writes a 32-bit integer value with variable length encoding and a specified block size.
- void [WriteUInt32VarLength](#) (uint value)
Writes a 32-bit unsigned integer value with variable length encoding.
- void [WriteUInt32VarLength](#) (uint value, int blockSize)
Writes a 32-bit unsigned integer value with variable length encoding and a specified block size.
- void [WriteUInt64VarLength](#) (ulong value, int blockSize)
Writes a 64-bit unsigned integer value with variable length encoding and a specified block size.

Public Attributes

- int [_offsetBits](#)

Properties

- int [OffsetBits](#) [get, set]
Gets or sets the offset in bits.

6.330.1 Detailed Description

Represents a null bit buffer for writing data.

6.330.2 Member Function Documentation

6.330.2.1 PadToByteBoundary()

```
void PadToByteBoundary ( )
```

Pads the buffer to the next byte boundary by writing zero bits if necessary.

6.330.2.2 WriteBoolean()

```
bool WriteBoolean (
    bool b )
```

Writes a boolean value to the buffer and increments the offset by 1 bit.

Parameters

<i>b</i>	The boolean value to write.
----------	-----------------------------

Returns

The boolean value that was written.

Implements [INetBitWriteStream](#).

6.330.2.3 WriteByte()

```
void WriteByte (
    byte value,
    int bits = 8 )
```

Writes a byte value to the buffer and increments the offset by the specified number of bits.

Parameters

<i>value</i>	The byte value to write.
<i>bits</i>	The number of bits to write. Default is 8.

6.330.2.4 WriteBytesAligned()

```
unsafe void WriteBytesAligned (
    void * buffer,
    int length )
```

Writes a specified number of bytes from a buffer to the bit buffer, ensuring byte alignment.

Parameters

<i>buffer</i>	A pointer to the buffer containing the bytes to write.
<i>length</i>	The number of bytes to write.

Implements [INetBitWriteStream](#).

6.330.2.5 WriteInt32()

```
void WriteInt32 (
    int value,
    int bits = 32 )
```

Writes a 32-bit integer value to the buffer and increments the offset by the specified number of bits.

Parameters

<i>value</i>	The 32-bit integer value to write.
<i>bits</i>	The number of bits to write. Default is 32.

Implements [INetBitWriteStream](#).

6.330.2.6 WriteInt32VarLength() [1/2]

```
void WriteInt32VarLength (
    int value )
```

Writes a 32-bit integer value with variable length encoding.

Parameters

<i>value</i>	The 32-bit integer value to write.
--------------	------------------------------------

Implements [INetBitWriteStream](#).

6.330.2.7 WriteInt32VarLength() [2/2]

```
void WriteInt32VarLength (
    int value,
    int blockSize )
```

Writes a 32-bit integer value with variable length encoding and a specified block size.

Parameters

<i>value</i>	The 32-bit integer value to write.
<i>blockSize</i>	The block size for encoding.

Implements [INetBitWriteStream](#).

6.330.2.8 WriteUInt32VarLength() [1/2]

```
void WriteUInt32VarLength (
    uint value )
```

Writes a 32-bit unsigned integer value with variable length encoding.

Parameters

<i>value</i>	The 32-bit unsigned integer value to write.
--------------	---

6.330.2.9 WriteUInt32VarLength() [2/2]

```
void WriteUInt32VarLength (
    uint value,
    int blockSize )
```

Writes a 32-bit unsigned integer value with variable length encoding and a specified block size.

Parameters

<i>value</i>	The 32-bit unsigned integer value to write.
<i>blockSize</i>	The block size for encoding.

6.330.2.10 WriteUInt64VarLength()

```
void WriteUInt64VarLength (
    ulong value,
    int blockSize )
```

Writes a 64-bit unsigned integer value with variable length encoding and a specified block size.

Parameters

<i>value</i>	The 64-bit unsigned integer value to write.
<i>blockSize</i>	The block size for encoding.

Implements [INetBitWriteStream](#).

6.330.3 Property Documentation**6.330.3.1 OffsetBits**

```
int OffsetBits [get], [set]
```

Gets or sets the offset in bits.

The current offset in bits.

6.331 NetBitBufferSerializer Struct Reference

Represents a serializer for reading and writing data to a [NetBitBuffer](#).

Public Member Functions

- bool [Check](#) (bool value)
Checks the boolean value and writes or reads it from the buffer based on the serializer mode.
- **NetBitBufferSerializer** ([NetBitBuffer](#) *buffer, bool write)
- void [Serialize](#) (ref bool value)
Serializes a boolean value by writing or reading it from the buffer based on the serializer mode.
- void [Serialize](#) (ref byte value)
Serializes a byte value by writing or reading it from the buffer based on the serializer mode.
- void [Serialize](#) (ref float value)
Serializes a float value by writing or reading it from the buffer based on the serializer mode.
- void [Serialize](#) (ref int value)
Serializes an integer value by writing or reading it from the buffer based on the serializer mode.
- void [Serialize](#) (ref string value)
Serializes a string value by writing or reading it from the buffer based on the serializer mode.
- void [Serialize](#) (ref uint value)
Serializes an unsigned integer value by writing or reading it from the buffer based on the serializer mode.
- void [Serialize](#) (ref ulong value)
Serializes an unsigned long value by writing or reading it from the buffer based on the serializer mode.

Static Public Member Functions

- static [NetBitBufferSerializer Reader](#) ([NetBitBuffer](#) *buffer)
Creates a new instance of [NetBitBufferSerializer](#) for reading from the buffer.
- static [NetBitBufferSerializer Writer](#) ([NetBitBuffer](#) *buffer)
Creates a new instance of [NetBitBufferSerializer](#) for writing to the buffer.

Public Attributes

- [NetBitBuffer](#) * **_buffer**
- bool **_write**

Properties

- [NetBitBuffer](#) * **Buffer** [get]
Gets the buffer associated with the serializer.
- bool **Reading** [get]
Gets a value indicating whether the serializer is in read mode.
- bool **Writing** [get]
Gets a value indicating whether the serializer is in write mode.

6.331.1 Detailed Description

Represents a serializer for reading and writing data to a [NetBitBuffer](#).

6.331.2 Member Function Documentation

6.331.2.1 Check()

```
bool Check (  
    bool value )
```

Checks the boolean value and writes or reads it from the buffer based on the serializer mode.

Parameters

<i>value</i>	The boolean value to check.
--------------	-----------------------------

Returns

The boolean value read from the buffer if in read mode, otherwise the value written to the buffer.

6.331.2.2 Reader()

```
static NetBitBufferSerializer Reader (  
    NetBitBuffer * buffer ) [static]
```

Creates a new instance of [NetBitBufferSerializer](#) for reading from the buffer.

Parameters

<i>buffer</i>	The buffer to read from.
---------------	--------------------------

Returns

A new instance of [NetBitBufferSerializer](#) configured for reading.

6.331.2.3 Serialize() [1/7]

```
void Serialize (  
    ref bool value )
```

Serializes a boolean value by writing or reading it from the buffer based on the serializer mode.

Parameters

<i>value</i>	The boolean value to serialize.
--------------	---------------------------------

6.331.2.4 Serialize() [2/7]

```
void Serialize (  
    ref byte value )
```

Serializes a byte value by writing or reading it from the buffer based on the serializer mode.

Parameters

<i>value</i>	The byte value to serialize.
--------------	------------------------------

6.331.2.5 Serialize() [3/7]

```
void Serialize (  
    ref float value )
```

Serializes a float value by writing or reading it from the buffer based on the serializer mode.

Parameters

<i>value</i>	The float value to serialize.
--------------	-------------------------------

6.331.2.6 Serialize() [4/7]

```
void Serialize (  
    ref int value )
```

Serializes an integer value by writing or reading it from the buffer based on the serializer mode.

Parameters

<i>value</i>	The integer value to serialize.
--------------	---------------------------------

6.331.2.7 Serialize() [5/7]

```
void Serialize (  
    ref string value )
```

Serializes a string value by writing or reading it from the buffer based on the serializer mode.

Parameters

<i>value</i>	The string value to serialize.
--------------	--------------------------------

6.331.2.8 Serialize() [6/7]

```
void Serialize (
    ref uint value )
```

Serializes an unsigned integer value by writing or reading it from the buffer based on the serializer mode.

Parameters

<i>value</i>	The unsigned integer value to serialize.
--------------	--

6.331.2.9 Serialize() [7/7]

```
void Serialize (
    ref ulong value )
```

Serializes an unsigned long value by writing or reading it from the buffer based on the serializer mode.

Parameters

<i>value</i>	The unsigned long value to serialize.
--------------	---------------------------------------

6.331.2.10 Writer()

```
static NetBitBufferSerializer Writer (
    NetBitBuffer * buffer ) [static]
```

Creates a new instance of [NetBitBufferSerializer](#) for writing to the buffer.

Parameters

<i>buffer</i>	The buffer to write to.
---------------	-------------------------

Returns

A new instance of [NetBitBufferSerializer](#) configured for writing.

6.331.3 Property Documentation

6.331.3.1 Buffer

```
NetBitBuffer* Buffer [get]
```

Gets the buffer associated with the serializer.

6.331.3.2 Reading

```
bool Reading [get]
```

Gets a value indicating whether the serializer is in read mode.

6.331.3.3 Writing

```
bool Writing [get]
```

Gets a value indicating whether the serializer is in write mode.

6.332 NetCommandAccepted Struct Reference

Accepted Command, sent by the server when a remote client connection is accepted

Static Public Member Functions

- static [NetCommandAccepted](#) **Create** ([NetConnectionId](#) localId, [NetConnectionId](#) remoteId, uint counter)

Public Attributes

- [NetConnectionId](#) **AcceptedLocalId**
- [NetConnectionId](#) **AcceptedRemoteId**
- uint **Counter**
- [NetCommandHeader](#) **Header**

6.332.1 Detailed Description

Accepted Command, sent by the server when a remote client connection is accepted

6.333 NetCommandConnect Struct Reference

Connect Command used to signal a remote server that a client is trying to connect to it

Static Public Member Functions

- static int **ClampTokenLength** (int tokenLength)
- static [NetCommandConnect](#) **Create** ([NetConnectionId](#) id, byte *token=null, int tokenLength=0, byte *uniqueId=null)
- static byte[] **GetTokenDataAsArray** ([NetCommandConnect](#) command)
- static byte[] **GetUniqueIdAsArray** ([NetCommandConnect](#) command)

Public Attributes

- [NetConnectionId](#) **ConnectionId**
- [NetCommandHeader](#) **Header**
- fixed byte **TokenData** [TOKEN_MAX_LENGTH_BYTES]
- int **TokenLength**
- fixed byte **UniqueId** [UNIQUE_ID_LENGTH_BYTES]

Static Public Attributes

- const int **SIZE_BITS** = SIZE_BYTES * 8
- const int **SIZE_BYTES** = 16 + TOKEN_MAX_LENGTH_BYTES + UNIQUE_ID_LENGTH_BYTES
- const int **TOKEN_MAX_LENGTH_BYTES** = 128
- const int **UNIQUE_ID_LENGTH_BYTES** = NetConnection.UNIQUE_ID_SIZE

6.333.1 Detailed Description

Connect Command used to signal a remote server that a client is trying to connect to it

6.334 NetCommandDisconnect Struct Reference

Disconnect Command, it can be used by either side of the connection

Static Public Member Functions

- static [NetCommandDisconnect](#) **Create** ([NetDisconnectReason](#) reason, byte *token, int tokenLength)
- static [NetCommandDisconnect](#) **Create** ([NetDisconnectReason](#) reason, byte[] token)

Public Attributes

- [NetCommandHeader](#) **Header**
- [NetDisconnectReason](#) **Reason**
- fixed byte **TokenData** [TOKEN_MAX_LENGTH_BYTES]
- int **TokenLength**

Static Public Attributes

- const int **TOKEN_MAX_LENGTH_BYTES** = 128

6.334.1 Detailed Description

Disconnect Command, it can be used by either side of the connection

6.335 NetCommandHeader Struct Reference

Network Command Header Describe its type and usual settings for all commands

Static Public Member Functions

- static [NetCommandHeader Create](#) ([NetCommands](#) command)
Create a new [NetCommandHeader](#) based on a [NetCommands](#) type
- static implicit **operator [NetCommandHeader](#)** ([NetCommands](#) commands)

Public Attributes

- [NetCommands](#) **Command**
- [NetPacketType](#) **PacketType**

Static Public Attributes

- const int **SIZE_BITS** = SIZE_BYTES * 8
- const int **SIZE_BYTES** = 2

6.335.1 Detailed Description

Network Command Header Describe its type and usual settings for all commands

6.335.2 Member Function Documentation

6.335.2.1 Create()

```
static NetCommandHeader Create (  
    NetCommands command ) [static]
```

Create a new [NetCommandHeader](#) based on a [NetCommands](#) type

Parameters

<i>command</i>	Type of Command that should be created
----------------	--

Returns

New [NetCommandHeader](#) reference based on the Command Type

6.336 NetCommandRefused Struct Reference

Refuse Command, sent by the server when the connection was refused. This happens when the server has reached its max connection capacity.

Static Public Member Functions

- static [NetCommandRefused](#) **Create** ([NetConnectFailedReason](#) reason)

Public Attributes

- [NetCommandHeader](#) **Header**
- [NetConnectFailedReason](#) **Reason**

Static Public Attributes

- const int **SIZE_IN_BITS** = SIZE_IN_BYTES * 8
- const int **SIZE_IN_BYTES** = 3

6.336.1 Detailed Description

Refuse Command, sent by the server when the connection was refused. This happens when the server has reached its max connection capacity.

6.337 NetConfig Struct Reference

General configuration used to drive the behavior of the Socket library

Public Attributes

- [NetAddress Address](#)
Network Address used to bind the internal Socket
- int [ConnectAttempts](#)
Number of Connection Attempts tried by the peer before cancel the connection
- double [ConnectInterval](#)
Interval in Seconds between attempts to connect to a remote server
- double [ConnectionDefaultRtt](#)
Initial RTT
- int [ConnectionGroups](#)
Number of Connection Groups supported by the local instance
- double [ConnectionPingInterval](#)
Interval in Seconds between ping being sent to a remote end
- int [ConnectionSendBuffers](#)
Pre-allocated number of data buffers used to send data
- double [ConnectionShutdownTime](#)
Timeout in Seconds to allow a disconnected Connection to be released from the Group Mapping
- double [ConnectionTimeout](#)
Connection Timeout in seconds
- int [MaxConnections](#)
Max Number of Connections supported by the local instance
- [NetConfigNotify Notify](#)
Package acknowledgment system configuration
- double [OperationExpireTime](#)
Max Allowed time for the Send and Receive operations, in milliseconds
- int [PacketSize](#)
UDP Packet Size in Bytes
- [NetConfigSimulation Simulation](#)
Network simulation system configuration
- int [SocketRecvBuffer](#)
Size of the internal Socket receive buffer
- int [SocketSendBuffer](#)
Size of the internal Socket send buffer

Properties

- int [ConnectionsPerGroup](#) [get]
Max number of Connection per Group based on the [ConnectionGroups](#) and [MaxConnections](#)
- static [NetConfig Defaults](#) [get]
Builds a [NetConfig](#) with the default values
- int [PacketSizeInBits](#) [get]
UDP Packet Size in Bits based on [PacketSize](#)

6.337.1 Detailed Description

General configuration used to drive the behavior of the Socket library

6.337.2 Member Data Documentation

6.337.2.1 Address

`NetAddress` Address

Network Address used to bind the internal Socket

6.337.2.2 ConnectAttempts

`int` ConnectAttempts

Number of Connection Attempts tried by the peer before cancel the connection

6.337.2.3 ConnectInterval

`double` ConnectInterval

Interval in Seconds between attempts to connect to a remote server

6.337.2.4 ConnectionDefaultRtt

`double` ConnectionDefaultRtt

Initial RTT

6.337.2.5 ConnectionGroups

`int` ConnectionGroups

Number of Connection Groups supported by the local instance

6.337.2.6 ConnectionPingInterval

```
double ConnectionPingInterval
```

Interval in Seconds between ping being sent to a remote end

6.337.2.7 ConnectionSendBuffers

```
int ConnectionSendBuffers
```

Pre-allocated number of data buffers used to send data

6.337.2.8 ConnectionShutdownTime

```
double ConnectionShutdownTime
```

Timeout in Seconds to allow a disconnected Connection to be released from the Group Mapping

NetPeerGroup.UpdateShutdown

6.337.2.9 ConnectionTimeout

```
double ConnectionTimeout
```

Connection Timeout in seconds

6.337.2.10 MaxConnections

```
int MaxConnections
```

Max Number of Connections supported by the local instance

6.337.2.11 Notify

```
NetConfigNotify Notify
```

Package acknowledgment system configuration

6.337.2.12 OperationExpireTime

`double OperationExpireTime`

Max Allowed time for the Send and Receive operations, in milliseconds

6.337.2.13 PacketSize

`int PacketSize`

UDP Packet Size in Bytes

6.337.2.14 Simulation

`NetConfigSimulation Simulation`

Network simulation system configuration

6.337.2.15 SocketRecvBuffer

`int SocketRecvBuffer`

Size of the internal Socket receive buffer

6.337.2.16 SocketSendBuffer

`int SocketSendBuffer`

Size of the internal Socket send buffer

6.337.3 Property Documentation

6.337.3.1 ConnectionsPerGroup

`int ConnectionsPerGroup [get]`

Max number of Connection per Group based on the [ConnectionGroups](#) and [MaxConnections](#)

6.337.3.2 Defaults

`NetConfig Defaults` [static], [get]

Builds a `NetConfig` with the default values

6.337.3.3 PacketSizeInBits

`int PacketSizeInBits` [get]

UDP Packet Size in Bits based on `PacketSize`

6.338 NetConfigNotify Struct Reference

Represents the configuration for network notifications.

Public Attributes

- int `AckForceCount`
The count of forced acknowledgments.
- double `AckForceTimeout`
The timeout for forced acknowledgments.
- int `AckMaskBytes`
The number of bytes used for acknowledgment masks.
- int `SequenceBytes`
The number of bytes used for sequences.
- int `WindowSize`
The size of the window.

Properties

- int `AckMaskBits` [get]
Gets the number of bits used for acknowledgment masks.
- static `NetConfigNotify Defaults` [get]
Gets the default configuration for network notifications.
- int `SequenceBounds` [get]
*Gets the sequence bounds, calculated as $WindowSize * 16$.*

6.338.1 Detailed Description

Represents the configuration for network notifications.

6.338.2 Member Data Documentation

6.338.2.1 AckForceCount

```
int AckForceCount
```

The count of forced acknowledgments.

6.338.2.2 AckForceTimeout

```
double AckForceTimeout
```

The timeout for forced acknowledgments.

6.338.2.3 AckMaskBytes

```
int AckMaskBytes
```

The number of bytes used for acknowledgment masks.

6.338.2.4 SequenceBytes

```
int SequenceBytes
```

The number of bytes used for sequences.

6.338.2.5 WindowSize

```
int WindowSize
```

The size of the window.

6.338.3 Property Documentation

6.338.3.1 AckMaskBits

```
int AckMaskBits [get]
```

Gets the number of bits used for acknowledgment masks.

6.338.3.2 Defaults

```
NetConfigNotify Defaults [static], [get]
```

Gets the default configuration for network notifications.

6.338.3.3 SequenceBounds

```
int SequenceBounds [get]
```

Gets the sequence bounds, calculated as WindowSize * 16.

6.339 NetConfigSimulation Struct Reference

Represents the configuration for network simulation.

Static Public Member Functions

- static [NetConfigSimulation WithLossNotifySequences](#) (params short[] sequences)
Creates a configuration with specified loss notification sequences.

Public Attributes

- [NetConfigSimulationOscillator DelayOscillator](#)
The oscillator used for simulating delay.
- double [DuplicateChance](#)
The chance of a duplicate packet.
- short * [LossNotifySequences](#)
Pointer to the sequences used for loss notifications.
- int [LossNotifySequencesLength](#)
The length of the loss notification sequences.
- [NetConfigSimulationOscillator LossOscillator](#)
The oscillator used for simulating loss.

Properties

- static [NetConfigSimulation Defaults](#) [get]
Gets the default configuration for network simulation.

6.339.1 Detailed Description

Represents the configuration for network simulation.

6.339.2 Member Function Documentation

6.339.2.1 WithLossNotifySequences()

```
static NetConfigSimulation WithLossNotifySequences (  
    params short[] sequences ) [static]
```

Creates a configuration with specified loss notification sequences.

Parameters

<i>sequences</i>	The sequences to be used for loss notifications.
------------------	--

Returns

A new instance of [NetConfigSimulation](#) with the specified sequences.

6.339.3 Member Data Documentation

6.339.3.1 DelayOscillator

```
NetConfigSimulationOscillator DelayOscillator
```

The oscillator used for simulating delay.

6.339.3.2 DuplicateChance

```
double DuplicateChance
```

The chance of a duplicate packet.

6.339.3.3 LossNotifySequences

```
short* LossNotifySequences
```

Pointer to the sequences used for loss notifications.

6.339.3.4 LossNotifySequencesLength

```
int LossNotifySequencesLength
```

The length of the loss notification sequences.

6.339.3.5 LossOscillator

```
NetConfigSimulationOscillator LossOscillator
```

The oscillator used for simulating loss.

6.339.4 Property Documentation

6.339.4.1 Defaults

```
NetConfigSimulation Defaults [static], [get]
```

Gets the default configuration for network simulation.

6.340 NetConfigSimulationOscillator Struct Reference

Represents an oscillator configuration for network simulation.

Public Types

- enum class [WaveShape](#)
Defines the shape of the wave.

Public Member Functions

- double [GetCurveValue](#) (System.Random rng, double elapsedSecs)
Calculates the value of the wave at a given time.

Public Attributes

- double [Additional](#)
Additional noise to be added to the wave.
- double [Max](#)
The maximum value of the wave.
- double [Min](#)
The minimum value of the wave.
- double [Period](#)
The period of the wave in seconds.
- [WaveShape Shape](#)
The shape of the wave.
- double [Threshold](#)
The threshold value for the wave.

6.340.1 Detailed Description

Represents an oscillator configuration for network simulation.

6.340.2 Member Enumeration Documentation

6.340.2.1 WaveShape

```
enum WaveShape [strong]
```

Defines the shape of the wave.

Enumerator

Noise	Random noise wave shape.
Sine	Sine wave shape.
Square	Square wave shape.
Triangle	Triangle wave shape.
Saw	Sawtooth wave shape.
ReverseSaw	Reverse sawtooth wave shape.

6.340.3 Member Function Documentation

6.340.3.1 GetCurveValue()

```
double GetCurveValue (
    System.Random rng,
    double elapsedSecs )
```

Calculates the value of the wave at a given time.

Parameters

<i>rng</i>	The random number generator.
<i>elapsedSecs</i>	The elapsed time in seconds.

Returns

The calculated wave value.

6.340.4 Member Data Documentation

6.340.4.1 Additional

`double Additional`

Additional noise to be added to the wave.

6.340.4.2 Max

`double Max`

The maximum value of the wave.

6.340.4.3 Min

`double Min`

The minimum value of the wave.

6.340.4.4 Period

`double Period`

The period of the wave in seconds.

6.340.4.5 Shape

[WaveShape](#) Shape

The shape of the wave.

6.340.4.6 Threshold

double Threshold

The threshold value for the wave.

6.341 NetConnection Struct Reference

Network connection

Public Member Functions

- override string [ToString](#) ()
Returns a string that represents the current [NetConnection](#).

Properties

- bool [Active](#) [get]
Gets a value indicating whether the connection is active.
- [NetConnectionStatus](#) [ConnectionStatus](#) [get]
Gets the current status of the connection.
- [NetConnectionId](#) [LocalConnectionId](#) [get]
Gets the local connection ID.
- [NetAddress](#) [RemoteAddress](#) [get]
Gets the remote address of the connection.
- [NetConnectionId](#) [RemoteConnectionId](#) [get]
Gets the remote connection ID.
- double [RoundTripTime](#) [get]
Gets the round-trip time (RTT) for the connection.

6.341.1 Detailed Description

Network connection

6.341.2 Member Function Documentation

6.341.2.1 ToString()

```
override string ToString ( )
```

Returns a string that represents the current [NetConnection](#).

6.341.3 Property Documentation

6.341.3.1 Active

```
bool Active [get]
```

Gets a value indicating whether the connection is active.

6.341.3.2 ConnectionStatus

```
NetConnectionStatus ConnectionStatus [get]
```

Gets the current status of the connection.

6.341.3.3 LocalConnectionId

```
NetConnectionId LocalConnectionId [get]
```

Gets the local connection ID.

6.341.3.4 RemoteAddress

```
NetAddress RemoteAddress [get]
```

Gets the remote address of the connection.

6.341.3.5 RemoteConnectionId

```
NetConnectionId RemoteConnectionId [get]
```

Gets the remote connection ID.

6.341.3.6 RoundTripTime

```
double RoundTripTime [get]
```

Gets the round-trip time (RTT) for the connection.

6.342 NetConnectionId Struct Reference

Represents a network connection ID.

Inherits `IEquatable< NetConnectionId >`.

Classes

- class [EqualityComparer](#)
An equality comparer for [NetConnectionId](#) instances.

Public Member Functions

- bool [Equals](#) ([NetConnectionId](#) other)
Compares this [NetConnectionId](#) to another for equality.
- override bool [Equals](#) (object obj)
Compares this [NetConnectionId](#) to another for equality.
- override int [GetHashCode](#) ()
Get the hash code for this [NetConnectionId](#).

Static Public Member Functions

- static bool [operator!=](#) ([NetConnectionId](#) a, [NetConnectionId](#) b)
Compares two [NetConnectionId](#) instances for inequality.
- static bool [operator==](#) ([NetConnectionId](#) a, [NetConnectionId](#) b)
Compares two [NetConnectionId](#) instances for equality.

Public Attributes

- short [Group](#)
The group of the connection.
- short [GroupIndex](#)
The index of the connection within the group.

6.342.1 Detailed Description

Represents a network connection ID.

6.342.2 Member Function Documentation

6.342.2.1 Equals() [1/2]

```
bool Equals (  
    NetConnectionId other )
```

Compares this [NetConnectionId](#) to another for equality.

6.342.2.2 Equals() [2/2]

```
override bool Equals (  
    object obj )
```

Compares this [NetConnectionId](#) to another for equality.

6.342.2.3 GetHashCode()

```
override int GetHashCode ( )
```

Get the hash code for this [NetConnectionId](#).

6.342.2.4 operator"!="()

```
static bool operator!= (  
    NetConnectionId a,  
    NetConnectionId b ) [static]
```

Compares two [NetConnectionId](#) instances for inequality.

6.342.2.5 operator=="()

```
static bool operator== (  
    NetConnectionId a,  
    NetConnectionId b ) [static]
```

Compares two [NetConnectionId](#) instances for equality.

6.342.3 Member Data Documentation

6.342.3.1 Group

short Group

The group of the connection.

6.342.3.2 GroupIndex

short GroupIndex

The index of the connection within the group.

6.343 NetConnectionId.EqualityComparer Class Reference

An equality comparer for [NetConnectionId](#) instances.

Inherits `IEqualityComparer< NetConnectionId >`.

Public Member Functions

- bool [Equals](#) ([NetConnectionId](#) x, [NetConnectionId](#) y)
Compares two [NetConnectionId](#) instances for equality.
- int [GetHashCode](#) ([NetConnectionId](#) obj)
Get the hash code for the [NetConnectionId](#).

6.343.1 Detailed Description

An equality comparer for [NetConnectionId](#) instances.

6.343.2 Member Function Documentation

6.343.2.1 Equals()

```
bool Equals (  
    NetConnectionId x,  
    NetConnectionId y )
```

Compares two [NetConnectionId](#) instances for equality.

6.343.2.2 GetHashCode()

```
int GetHashCode (
    NetConnectionId obj )
```

Get the hash code for the [NetConnectionId](#).

6.344 NetConnectionMap Struct Reference

Represents a network connection map.

Classes

- struct [Iterator](#)
Iterator for traversing the connections in a [NetConnectionMap](#).

Public Types

- enum class [EntryState](#)
Represents the state of an entry in the [NetConnectionMap](#).

Public Member Functions

- [NetConnection](#) * [Find](#) ([NetAddress](#) address)
Finds a connection by its address in the network peer group.
- [NetConnection](#) * [Find](#) ([NetConnectionId](#) id)
Finds a connection by its ID in the network peer group.
- [NetConnection](#) * [FindByIndex](#) (int index)
Gets a connection by its index in the network peer group.
- [NetConnection](#) * [Insert](#) ([NetAddress](#) address, byte[] uniqueId)
Inserts a new connection into the network peer group using the specified address and unique ID.
- [NetConnection](#) * [Remap](#) ([NetAddress](#) oldAddress, [NetAddress](#) newAddress)
Remaps a connection from an old address to a new address.
- bool [Remove](#) ([NetAddress](#) address)
Removes a connection from the network peer group by its address.
- bool [TryFindByIndex](#) (int index, out [NetConnection](#) *connection)
Tries to get a connection by its index in the network peer group.

Static Public Member Functions

- static [NetConnectionMap](#) * [Allocate](#) (int capacity, short groupIndex, in [NetConfig](#) *config)
Allocates and initializes a new [NetConnectionMap](#) with the specified capacity and group index.
- static void [Dispose](#) ([NetConnectionMap](#) *map, [INetPeerGroupCallbacks](#) callbacks)
Disposes of the specified [NetConnectionMap](#) and its associated resources.

Public Attributes

- [NetConnection](#) ** **Buckets**
- `ulong` **CapacityAllocated**
- `ulong` **FreeCount**
- [NetConnection](#) * **FreeHead**
- `short` **Group**
- `ulong` **IdsCount**
- `UniqueldMapping` * **UniqueldHashes**
- `ulong` **UsedCount**

Properties

- [NetConnection](#) * [ConnectionsBuffer](#) [get]
Gets the buffer of connections.
- `int` [Count](#) [get]
Gets the count of used connections minus the free connections.
- `int` [CountUsed](#) [get]
Gets the count of used connections.
- `bool` [Full](#) [get]
Gets a value indicating whether the connection map is full.

6.344.1 Detailed Description

Represents a network connection map.

6.344.2 Member Enumeration Documentation

6.344.2.1 EntryState

`enum` [EntryState](#) [strong]

Represents the state of an entry in the [NetConnectionMap](#).

Enumerator

None	The entry is not in use.
Free	The entry is free and available for use.
Used	The entry is currently in use.

6.344.3 Member Function Documentation

6.344.3.1 Allocate()

```
static NetConnectionMap* Allocate (
    int capacity,
    short groupIndex,
    in NetConfig * config ) [static]
```

Allocates and initializes a new [NetConnectionMap](#) with the specified capacity and group index.

Parameters

<i>capacity</i>	The number of connections the map can hold.
<i>groupIndex</i>	The group index associated with the connections.
<i>config</i>	The configuration settings for the network connections.

Returns

A pointer to the newly allocated [NetConnectionMap](#).

6.344.3.2 Dispose()

```
static void Dispose (
    NetConnectionMap * map,
    INetPeerGroupCallbacks callbacks ) [static]
```

Disposes of the specified [NetConnectionMap](#) and its associated resources.

Parameters

<i>map</i>	A pointer to the NetConnectionMap to dispose.
<i>callbacks</i>	The callbacks to invoke during disposal.

6.344.3.3 Find() [1/2]

```
NetConnection* Find (
    NetAddress address )
```

Finds a connection by its address in the network peer group.

Parameters

<i>address</i>	The address of the connection to find.
----------------	--

Returns

A pointer to the connection if found, otherwise null.

6.344.3.4 Find() [2/2]

```
NetConnection* Find (  
    NetConnectionId id )
```

Finds a connection by its ID in the network peer group.

Parameters

<i>id</i>	The ID of the connection to find.
-----------	-----------------------------------

Returns

A pointer to the connection if found, otherwise null.

6.344.3.5 FindByIndex()

```
NetConnection* FindByIndex (  
    int index )
```

Gets a connection by its index in the network peer group.

Parameters

<i>index</i>	The index of the connection to retrieve.
--------------	--

Returns

A pointer to the connection if found, otherwise throws an `IndexOutOfRangeException`.

Exceptions

<i>IndexOutOfRangeException</i>	Thrown when the index is out of the usable capacity range.
---------------------------------	--

6.344.3.6 Insert()

```
NetConnection* Insert (  
    NetAddress address,  
    byte[] uniqueId )
```

Inserts a new connection into the network peer group using the specified address and unique ID.

Parameters

<i>address</i>	The address of the connection to insert.
<i>uniqueId</i>	The unique ID of the connection to insert.

Returns

A pointer to the newly inserted connection, or null if the connection is already in use or the capacity is full.

6.344.3.7 Remap()

```
NetConnection* Remap (
    NetAddress oldAddress,
    NetAddress newAddress )
```

Remaps a connection from an old address to a new address.

Parameters

<i>oldAddress</i>	The old address of the connection.
<i>newAddress</i>	The new address of the connection.

Returns

A pointer to the remapped connection, or null if the remapping failed.

6.344.3.8 Remove()

```
bool Remove (
    NetAddress address )
```

Removes a connection from the network peer group by its address.

Parameters

<i>address</i>	The address of the connection to remove.
----------------	--

Returns

True if the connection was successfully removed, otherwise false.

6.344.3.9 TryFindByIndex()

```
bool TryFindByIndex (
    int index,
    out NetConnection * connection )
```

Tries to get a connection by its index in the network peer group.

Parameters

<i>index</i>	The index of the connection to retrieve.
<i>connection</i>	Output parameter that will point to the connection if found.

Returns

True if the connection was found, otherwise false.

6.344.4 Property Documentation

6.344.4.1 ConnectionsBuffer

```
NetConnection* ConnectionsBuffer [get]
```

Gets the buffer of connections.

6.344.4.2 Count

```
int Count [get]
```

Gets the count of used connections minus the free connections.

6.344.4.3 CountUsed

```
int CountUsed [get]
```

Gets the count of used connections.

6.344.4.4 Full

```
bool Full [get]
```

Gets a value indicating whether the connection map is full.

6.345 NetConnectionMap.Iterator Struct Reference

[Iterator](#) for traversing the connections in a [NetConnectionMap](#).

Public Member Functions

- [Iterator](#) ([NetConnectionMap](#) *map)
Initializes a new instance of the [Iterator](#) struct.
- bool [Next](#) ()
Advances the iterator to the next connection.

Public Attributes

- int [_count](#)
- int [_index](#)
- [NetConnectionMap](#) * [_map](#)

Properties

- [NetConnection](#) *? [Current](#) [get]
Gets the current connection in the iteration.
- bool [IsValid](#) [get]
Gets a value indicating whether the current index is valid.

6.345.1 Detailed Description

[Iterator](#) for traversing the connections in a [NetConnectionMap](#).

6.345.2 Constructor & Destructor Documentation

6.345.2.1 Iterator()

```
Iterator (  
    NetConnectionMap * map )
```

Initializes a new instance of the [Iterator](#) struct.

Parameters

<i>map</i>	The NetConnectionMap to iterate over.
------------	---

6.345.3 Member Function Documentation

6.345.3.1 Next()

```
bool Next ( )
```

Advances the iterator to the next connection.

Returns

True if the iterator was successfully advanced to the next connection, otherwise false.

6.345.4 Property Documentation

6.345.4.1 Current

```
NetConnection*? Current [get]
```

Gets the current connection in the iteration.

6.345.4.2 IsValid

```
bool IsValid [get]
```

Gets a value indicating whether the current index is valid.

6.346 NetPeer Struct Reference

Network Peer

Static Public Member Functions

- static void [Destroy](#) ([NetPeer](#) *p, [INetSocket](#) socket, [INetPeerGroupCallbacks](#) callbacks)
Destroys the specified [NetPeer](#) instance and releases associated resources.
- static void [DestroySocket](#) ([NetPeer](#) *p, [INetSocket](#) socket, [INetPeerGroupCallbacks](#) callbacks)
- static short [FindGroupWithLeastAssignedAddresses](#) ([NetPeer](#) *p)
- static [NetConfig](#) * [GetConfigPointer](#) ([NetPeer](#) *p)
Gets a pointer to the configuration settings of the specified [NetPeer](#).
- static [NetPeerGroup](#) * [GetGroup](#) ([NetPeer](#) *p, int index)
Gets a pointer to the [NetPeerGroup](#) at the specified index.
- static [NetPeer](#) * [Initialize](#) ([NetConfig](#) config, [INetSocket](#) socket)
Initializes a new instance of the [NetPeer](#) structure with the specified configuration and socket.
- static void [Initialize](#) ([NetPeer](#) *p, [NetConfig](#) config, [INetSocket](#) socket)
Initializes the specified [NetPeer](#) instance with the given configuration and socket.
- static void [Recv](#) ([NetPeer](#) *p, [INetSocket](#) socket, bool *work, System.Random rng)
Receives data for the specified [NetPeer](#) and indicates whether work was done.
- static void [Recv](#) ([NetPeer](#) *p, [INetSocket](#) socket, System.Random rng)
Receives data for the specified [NetPeer](#).
- static bool [RecvBufferAvailable](#) ([NetPeer](#) *p)
- static void [RecvBufferPushToGroup](#) ([NetPeer](#) *p, [INetSocket](#) socket, Random rng)
- static bool [RecvExpired](#) ([NetPeer](#) *p)
- static void [RecvInternal](#) ([NetPeer](#) *p, [INetSocket](#) socket, bool *work, System.Random rng)
- static void [RemapAddress](#) ([NetPeer](#) *p, [NetAddress](#) oldAddress, [NetAddress](#) newAddress)
Remaps the network address of the specified [NetPeer](#).
- static void [Send](#) ([NetPeer](#) *p, [INetSocket](#) socket)
Sends data for the specified [NetPeer](#).
- static void [Send](#) ([NetPeer](#) *p, [INetSocket](#) socket, bool *work)
Sends data for the specified [NetPeer](#) and indicates whether work was done.
- static void [SendFromStack](#) ([NetPeer](#) *p, [INetSocket](#) socket, bool *work)
- static void [SendInternal](#) ([NetPeer](#) *p, [INetSocket](#) socket, bool *work)
- static void [Update](#) ([NetPeer](#) *p, [INetSocket](#) socket, bool *work, System.Random rng)
Updates the state of the specified [NetPeer](#) and indicates whether work was done.
- static void [Update](#) ([NetPeer](#) *p, [INetSocket](#) socket, System.Random rng)
Updates the state of the specified [NetPeer](#).

Public Attributes

- [NetAddress](#) _address
- [NetConfig](#) _config
- byte * _fragmentBuffer
- [NetPeerGroup](#) * _groups
- int * _groupsAssigned
- [NetPeerGroupMap](#) * _groupsMap
- [NetBitBuffer](#) * _recv
- [NetBitBufferBlock](#) * _recvBlock
- [Timer](#) _recvTimer
- [NetCommandRefused](#) * _refusedCommand
- [NetBitBufferStack](#) _sendStack
- volatile int _state

Static Public Attributes

- const int `DEFAULT_HEADERS` = 40 + 8 + 96
IPv6 header: 40 bytes, UDP header: 8 bytes, Realtime Header: 96 bytes
- const int `MAX_MTU_BITS_PAYLOAD` = `MAX_MTU_BYTES_PAYLOAD` * 8
MaximumTransferUnit bits. (ipv6 header: 40 bytes, udp header: 8 bytes)
- const int `MAX_MTU_BYTES_PAYLOAD` = `MAX_MTU_BYTES_TOTAL` - `DEFAULT_HEADERS`
MaximumTransferUnit payload bytes.
- const int `MAX_MTU_BYTES_TOTAL` = 1280
MaximumTransferUnit total bytes.
- const int `MAX_PACKET_BYTES_PAYLOAD` = (`MAX_MTU_BYTES_PAYLOAD` - `NetNotifyHeader.SIZE_IN_BYTES`) * `FRAG_MAX_COUNT`
Max packet payload size in bytes. (Considering the NetNotifyHeader size for each fragment)
- const int `MAX_PACKET_BYTES_TOTAL` = `MAX_MTU_BYTES_TOTAL` * `FRAG_MAX_COUNT`
Max packet total size in bytes.
- const int `STATE_RUNNING` = 0
- const int `STATE_SHUTDOWN` = 2

Properties

- `NetAddress Address` [get]
Gets the network address of the peer.
- `NetConfig Config` [get]
Gets the configuration settings for the peer.
- int `GroupCount` [get]
Gets the number of connection groups.
- bool `IsShutdown` [get]
Gets a value indicating whether the peer is shut down.

6.346.1 Detailed Description

Network Peer

6.346.2 Member Function Documentation

6.346.2.1 Destroy()

```
static void Destroy (
    NetPeer * p,
    INetSocket socket,
    INetPeerGroupCallbacks callbacks ) [static]
```

Destroys the specified `NetPeer` instance and releases associated resources.

Parameters

<i>p</i>	A pointer to the NetPeer instance.
<i>socket</i>	The socket used for network communication.
<i>callbacks</i>	The callbacks for the NetPeerGroup .

6.346.2.2 GetConfigPointer()

```
static NetConfig* GetConfigPointer (
    NetPeer * p ) [static]
```

Gets a pointer to the configuration settings of the specified [NetPeer](#).

Parameters

<i>p</i>	A pointer to the NetPeer instance.
----------	--

Returns

A pointer to the [NetConfig](#) if the peer is not shut down; otherwise, null.

6.346.2.3 GetGroup()

```
static NetPeerGroup* GetGroup (
    NetPeer * p,
    int index ) [static]
```

Gets a pointer to the [NetPeerGroup](#) at the specified index.

Parameters

<i>p</i>	A pointer to the NetPeer instance.
<i>index</i>	The index of the NetPeerGroup .

Returns

A pointer to the [NetPeerGroup](#) if the peer is not shut down; otherwise, null.

6.346.2.4 Initialize() [1/2]

```
static NetPeer* Initialize (
    NetConfig config,
    INetSocket socket ) [static]
```

Initializes a new instance of the [NetPeer](#) structure with the specified configuration and socket.

Parameters

<i>config</i>	The configuration settings for the NetPeer .
<i>socket</i>	The socket used for network communication.

Returns

A pointer to the newly initialized [NetPeer](#) instance.

6.346.2.5 Initialize() [2/2]

```
static void Initialize (
    NetPeer * p,
    NetConfig config,
    INetSocket socket ) [static]
```

Initializes the specified [NetPeer](#) instance with the given configuration and socket.

Parameters

<i>p</i>	A pointer to the NetPeer instance.
<i>config</i>	The configuration settings for the NetPeer .
<i>socket</i>	The socket used for network communication.

6.346.2.6 Recv() [1/2]

```
static void Recv (
    NetPeer * p,
    INetSocket socket,
    bool * work,
    System.Random rng ) [static]
```

Receives data for the specified [NetPeer](#) and indicates whether work was done.

Parameters

<i>p</i>	A pointer to the NetPeer instance.
<i>socket</i>	The socket used for network communication.
<i>work</i>	A pointer to a boolean indicating whether work was done.
<i>rng</i>	A random number generator.

6.346.2.7 Recv() [2/2]

```
static void Recv (  
    NetPeer * p,  
    INetSocket socket,  
    System.Random rng ) [static]
```

Receives data for the specified [NetPeer](#).

Parameters

<i>p</i>	A pointer to the NetPeer instance.
<i>socket</i>	The socket used for network communication.
<i>rng</i>	A random number generator.

6.346.2.8 RemapAddress()

```
static void RemapAddress (  
    NetPeer * p,  
    NetAddress oldAddress,  
    NetAddress newAddress ) [static]
```

Remaps the network address of the specified [NetPeer](#).

Parameters

<i>p</i>	A pointer to the NetPeer instance.
<i>oldAddress</i>	The old network address.
<i>newAddress</i>	The new network address.

6.346.2.9 Send() [1/2]

```
static void Send (  
    NetPeer * p,  
    INetSocket socket ) [static]
```

Sends data for the specified [NetPeer](#).

Parameters

<i>p</i>	A pointer to the NetPeer instance.
<i>socket</i>	The socket used for network communication.

6.346.2.10 Send() [2/2]

```
static void Send (
    NetPeer * p,
    INetSocket socket,
    bool * work ) [static]
```

Sends data for the specified [NetPeer](#) and indicates whether work was done.

Parameters

<i>p</i>	A pointer to the NetPeer instance.
<i>socket</i>	The socket used for network communication.
<i>work</i>	A pointer to a boolean indicating whether work was done.

6.346.2.11 Update() [1/2]

```
static void Update (
    NetPeer * p,
    INetSocket socket,
    bool * work,
    System.Random rng ) [static]
```

Updates the state of the specified [NetPeer](#) and indicates whether work was done.

Parameters

<i>p</i>	A pointer to the NetPeer instance.
<i>socket</i>	The socket used for network communication.
<i>work</i>	A pointer to a boolean indicating whether work was done.
<i>rng</i>	A random number generator.

6.346.2.12 Update() [2/2]

```
static void Update (
    NetPeer * p,
    INetSocket socket,
    System.Random rng ) [static]
```

Updates the state of the specified [NetPeer](#).

Parameters

<i>p</i>	A pointer to the NetPeer instance.
<i>socket</i>	The socket used for network communication.
<i>rng</i>	A random number generator.

6.346.3 Member Data Documentation

6.346.3.1 DEFAULT_HEADERS

```
const int DEFAULT_HEADERS = 40 + 8 + 96 [static]
```

IPv6 header: 40 bytes, UDP header: 8 bytes, Realtime Header: 96 bytes

6.346.3.2 MAX_MTU_BITS_PAYLOAD

```
const int MAX_MTU_BITS_PAYLOAD = MAX_MTU_BYTES_PAYLOAD * 8 [static]
```

MaximumTransferUnit bits. (ipv6 header: 40 bytes, udp header: 8 bytes)

Same as [MAX_MTU_BYTES_PAYLOAD](#) but in bits.

6.346.3.3 MAX_MTU_BYTES_PAYLOAD

```
const int MAX_MTU_BYTES_PAYLOAD = MAX_MTU_BYTES_TOTAL - DEFAULT_HEADERS [static]
```

MaximumTransferUnit payload bytes.

The maximum number of bytes available for data(payload) in a single UDP packet.

6.346.3.4 MAX_MTU_BYTES_TOTAL

```
const int MAX_MTU_BYTES_TOTAL = 1280 [static]
```

MaximumTransferUnit total bytes.

The maximum size of bytes that can be sent in a single UDP packet.

6.346.3.5 MAX_PACKET_BYTES_PAYLOAD

```
const int MAX_PACKET_BYTES_PAYLOAD = (MAX_MTU_BYTES_PAYLOAD - NetNotifyHeader.SIZE_IN_BYTES) * FRAG_MAX_COUNT [static]
```

Max packet payload size in bytes. (Considering the NetNotifyHeader size for each fragment)

The maximum number of bytes available for data that can be fragmented into multiple packets of size [MAX_MTU_BYTES_PAYLOAD](#).

6.346.3.6 MAX_PACKET_BYTES_TOTAL

```
const int MAX_PACKET_BYTES_TOTAL = MAX_MTU_BYTES_TOTAL * FRAG_MAX_COUNT [static]
```

Max packet total size in bytes.

The maximum number of bytes that can be fragmented into multiple packets of size [MAX_MTU_BYTES_TOTAL](#).

6.346.4 Property Documentation

6.346.4.1 Address

```
NetAddress Address [get]
```

Gets the network address of the peer.

6.346.4.2 Config

```
NetConfig Config [get]
```

Gets the configuration settings for the peer.

6.346.4.3 GroupCount

```
int GroupCount [get]
```

Gets the number of connection groups.

6.346.4.4 IsShutdown

```
bool IsShutdown [get]
```

Gets a value indicating whether the peer is shut down.

6.347 NetPeerGroup Struct Reference

Network Peer Group.

Static Public Member Functions

- static [NetConnection](#) * **AllocateConnection** ([NetPeerGroup](#) *g, [NetAddress](#) address, byte[] token, byte[] uniqueId)
- static void **ChangeConnectionAddressDuringConnecting** ([NetPeerGroup](#) *g, [NetConnection](#) *c, [NetAddress](#) newAddress)

Changes the address of a connection during the connecting phase.
- static void **ChangeConnectionStatus** ([NetPeerGroup](#) *g, [INetPeerGroupCallbacks](#) cb, [NetConnection](#) *c, [NetConnectionStatus](#) status)
- static void **Connect** ([NetPeerGroup](#) *g, [NetAddress](#) address, byte[] token, byte[] uniqueId=null)

Connects to a specified address with an optional unique ID and token.
- static void **Connect** ([NetPeerGroup](#) *g, string ip, ushort port, byte[] token, byte[] uniqueId=null)

Connects to a specified IP address and port with an optional unique ID and token.
- static [NetConnectionMap.Iterator](#) **ConnectionIterator** ([NetPeerGroup](#) *g)

Gets an iterator for the connections in the network peer group.
- static void **Disconnect** ([NetPeerGroup](#) *g, [NetConnection](#) *c, byte[] token)

Disconnects a given connection from the network peer group with an optional token.
- static [NetConnection](#) * **GetConnection** ([NetPeerGroup](#) *g, [NetConnectionId](#) id)

Gets a connection by its ID in the network peer group.
- static [NetConnection](#) * **GetConnectionByIndex** ([NetPeerGroup](#) *g, int index)

Gets a connection by its index in the network peer group.
- static double **GetConnectionIdleTime** ([NetPeerGroup](#) *g, [NetConnection](#) *c)

Gets the idle time of a connection.
- static bool **GetConnectionSendBuffer** ([NetPeerGroup](#) *g, [NetConnection](#) *c, out [NetBitBuffer](#) *b)
- static bool **GetNotifyDataBuffer** ([NetPeerGroup](#) *g, [NetConnection](#) *c, out [NetBitBuffer](#) *b)

Gets a notify data buffer for a given connection in the network peer group.
- static bool **GetUnreliableDataBuffer** ([NetPeerGroup](#) *g, [NetConnection](#) *c, out [NetBitBuffer](#) *b)

Gets an unreliable data buffer for a given connection in the network peer group.
- static void **HandleCommandAccepted** ([NetPeerGroup](#) *g, [INetPeerGroupCallbacks](#) cb, [NetConnection](#) *c, [NetCommandAccepted](#) cmd)
- static void **HandleCommandConnect** ([NetPeerGroup](#) *g, [INetPeerGroupCallbacks](#) cb, [NetConnection](#) *c, [NetCommandConnect](#) cmd)
- static void **HandleCommandDisconnect** ([NetPeerGroup](#) *g, [INetPeerGroupCallbacks](#) cb, [NetConnection](#) *c, [NetCommandDisconnect](#) cmd)
- static void **HandleCommandRefused** ([NetPeerGroup](#) *g, [INetPeerGroupCallbacks](#) cb, [NetConnection](#) *c, [NetCommandRefused](#) cmd)
- static void **HandlePacket** ([NetPeerGroup](#) *g, [INetPeerGroupCallbacks](#) cb, [NetConnection](#) *c, [NetBitBuffer](#) *b)
- static void **HandlePacketAcks** ([NetPeerGroup](#) *g, [INetPeerGroupCallbacks](#) cb, [NetConnection](#) *c, [NetNotifyHeader](#) h)
- static void **HandlePacketCommand** ([NetPeerGroup](#) *g, [INetPeerGroupCallbacks](#) cb, [NetConnection](#) *c, [NetBitBuffer](#) *b)
- static void **HandlePacketNotifyAcks** ([NetPeerGroup](#) *g, [INetPeerGroupCallbacks](#) cb, [NetConnection](#) *c, [NetBitBuffer](#) *b)
- static void **HandlePacketNotifyData** ([NetPeerGroup](#) *g, [INetPeerGroupCallbacks](#) cb, [NetConnection](#) *c, [NetBitBuffer](#) *b)
- static void **HandlePacketNotifyData_Part2** ([NetNotifyHeader](#) header, int sequenceDistance, [NetPeerGroup](#) *g, [INetPeerGroupCallbacks](#) cb, [NetConnection](#) *c, [NetBitBuffer](#) *b)
- static void **HandlePacketUnconnected** ([NetPeerGroup](#) *g, [INetPeerGroupCallbacks](#) cb, [NetBitBuffer](#) *b)
- static void **HandlePacketUnreliableData** ([NetPeerGroup](#) *g, [INetPeerGroupCallbacks](#) cb, [NetConnection](#) *c, [NetBitBuffer](#) *b)
- static void **QueueAddressUnmap** ([NetPeerGroup](#) *g, [NetConnection](#) *c)
- static void **Receive** ([NetPeerGroup](#) *g, [INetPeerGroupCallbacks](#) cb)
- static void **ReleaseConnection** ([NetPeerGroup](#) *g, [INetPeerGroupCallbacks](#) cb, [NetConnection](#) *c)

- static void **Send** ([NetPeerGroup](#) *g, [NetConnection](#) *c, [NetBitBuffer](#) *b)
- static bool **SendCommand**< T > ([NetPeerGroup](#) *g, [NetConnection](#) *c, T cmd)
- static void **SendCommandConnect** ([NetPeerGroup](#) *g, [INetPeerGroupCallbacks](#) cb, [NetConnection](#) *c)
- static bool **SendCommandUnconnected**< T > ([NetPeerGroup](#) *g, [NetAddress](#) address, T cmd)
- static bool **SendNotifyDataBuffer** ([NetPeerGroup](#) *g, [NetConnection](#) *c, [NetBitBuffer](#) *b, void *userData)
Sends a notify data buffer for a given connection in the network peer group.
- static void **SendReliable** ([NetPeerGroup](#) *g, [NetConnection](#) *c, [ReliableId](#) rid, byte *data, int dataLength)
Sends reliable data for a given connection in the network peer group.
- static void **SendUnconnected** ([NetPeerGroup](#) *g, [NetBitBuffer](#) *b)
- static bool **SendUnconnectedData** ([NetPeerGroup](#) *g, [NetAddress](#) address, void *data, int dataLength)
Sends unconnected data to a specified address.
- static bool **SendUnreliableDataBuffer** ([NetPeerGroup](#) *g, [NetConnection](#) *c, [NetBitBuffer](#) *b)
Sends an unreliable data buffer for a given connection in the network peer group.
- static bool **TryGetConnectionByIndex** ([NetPeerGroup](#) *g, int index, out [NetConnection](#) *connection)
Tries to get a connection by its index in the network peer group.
- static void **Update** ([NetPeerGroup](#) *g, [INetPeerGroupCallbacks](#) cb)
Updates the network peer group by processing received data, handling timeouts, and managing connection retries.
- static void **UpdateConnected** ([NetPeerGroup](#) *g, [INetPeerGroupCallbacks](#) cb, [NetConnection](#) *c)
- static void **UpdateConnecting** ([NetPeerGroup](#) *g, [INetPeerGroupCallbacks](#) cb, [NetConnection](#) *c)
- static void **UpdateConnections** ([NetPeerGroup](#) *g, [INetPeerGroupCallbacks](#) cb)
- static void **UpdateDisconnected** ([NetPeerGroup](#) *g, [INetPeerGroupCallbacks](#) cb, [NetConnection](#) *c)
- static void **UpdateShutdown** ([NetPeerGroup](#) *g, [INetPeerGroupCallbacks](#) cb, [NetConnection](#) *c)

Public Attributes

- [Timer](#) _clock
- [NetConfig](#) _config
- [NetConnectionMap](#) * _connectionsMap
- uint _counter
- short _group
- [NetPeer](#) * _peer
- IntPtr _recvHead
- [NetBitBufferStack](#) _recvStack
- [NetBitBufferBlock](#) * _sendBlock
- IntPtr _sendHead

Static Public Attributes

- const double **RELIABLE_SEND_INTERVAL** = 0.05

Properties

- int [ConnectionCount](#) [get]
Gets the number of active connections.
- int [Group](#) [get]
Gets the group index.
- double [Time](#) [get]
Gets the elapsed time in seconds.

6.347.1 Detailed Description

Network Peer Group.

6.347.2 Member Function Documentation

6.347.2.1 ChangeConnectionAddressDuringConnecting()

```
static void ChangeConnectionAddressDuringConnecting (
    NetPeerGroup * g,
    NetConnection * c,
    NetAddress newAddress ) [static]
```

Changes the address of a connection during the connecting phase.

Parameters

<i>g</i>	Pointer to the network peer group.
<i>c</i>	Pointer to the connection.
<i>newAddress</i>	The new address to be assigned to the connection.

6.347.2.2 Connect() [1/2]

```
static void Connect (
    NetPeerGroup * g,
    NetAddress address,
    byte[] token,
    byte[] uniqueId = null ) [static]
```

Connects to a specified address with an optional unique ID and token.

Parameters

<i>g</i>	Pointer to the network peer group.
<i>address</i>	The address to connect to.
<i>token</i>	The token used for the connection.
<i>uniqueId</i>	Optional unique ID for the connection.

6.347.2.3 Connect() [2/2]

```
static void Connect (
```

```

    NetPeerGroup * g,
    string ip,
    ushort port,
    byte[] token,
    byte[] uniqueId = null ) [static]

```

Connects to a specified IP address and port with an optional unique ID and token.

Parameters

<i>g</i>	Pointer to the network peer group.
<i>ip</i>	The IP address to connect to.
<i>port</i>	The port to connect to.
<i>token</i>	The token used for the connection.
<i>uniqueId</i>	Optional unique ID for the connection.

6.347.2.4 ConnectionIterator()

```

static NetConnectionMap.Iterator ConnectionIterator (
    NetPeerGroup * g ) [static]

```

Gets an iterator for the connections in the network peer group.

Parameters

<i>g</i>	Pointer to the network peer group.
----------	------------------------------------

Returns

An iterator for the connections in the network peer group.

6.347.2.5 Disconnect()

```

static void Disconnect (
    NetPeerGroup * g,
    NetConnection * c,
    byte[] token ) [static]

```

Disconnects a given connection from the network peer group with an optional token.

Parameters

<i>g</i>	Pointer to the network peer group.
<i>c</i>	Pointer to the connection to be disconnected.
<i>token</i>	Optional token used for the disconnection.

6.347.2.6 GetConnection()

```
static NetConnection* GetConnection (
    NetPeerGroup * g,
    NetConnectionId id ) [static]
```

Gets a connection by its ID in the network peer group.

Parameters

<i>g</i>	Pointer to the network peer group.
<i>id</i>	The ID of the connection to retrieve.

Returns

A pointer to the connection if found, otherwise null.

6.347.2.7 GetConnectionByIndex()

```
static NetConnection* GetConnectionByIndex (
    NetPeerGroup * g,
    int index ) [static]
```

Gets a connection by its index in the network peer group.

Parameters

<i>g</i>	Pointer to the network peer group.
<i>index</i>	The index of the connection to retrieve.

Returns

A pointer to the connection if found, otherwise null.

6.347.2.8 GetConnectionIdleTime()

```
static double GetConnectionIdleTime (
    NetPeerGroup * g,
    NetConnection * c ) [static]
```

Gets the idle time of a connection.

Parameters

<i>g</i>	The network peer group.
<i>c</i>	The connection whose idle time is to be calculated.

Returns

The idle time of the connection in seconds.

6.347.2.9 GetNotifyDataBuffer()

```
static bool GetNotifyDataBuffer (
    NetPeerGroup * g,
    NetConnection * c,
    out NetBitBuffer * b ) [static]
```

Gets a notify data buffer for a given connection in the network peer group.

Parameters

<i>g</i>	Pointer to the network peer group.
<i>c</i>	Pointer to the connection.
<i>b</i>	Output parameter that will point to the bit buffer containing the data to be sent.

Returns

True if the buffer was successfully acquired, otherwise false.

6.347.2.10 GetUnreliableDataBuffer()

```
static bool GetUnreliableDataBuffer (
    NetPeerGroup * g,
    NetConnection * c,
    out NetBitBuffer * b ) [static]
```

Gets an unreliable data buffer for a given connection in the network peer group.

Parameters

<i>g</i>	Pointer to the network peer group.
<i>c</i>	Pointer to the connection.
<i>b</i>	Output parameter that will point to the bit buffer containing the data to be sent.

Returns

True if the buffer was successfully acquired, otherwise false.

6.347.2.11 SendNotifyDataBuffer()

```
static bool SendNotifyDataBuffer (  
    NetPeerGroup * g,  
    NetConnection * c,  
    NetBitBuffer * b,  
    void * userData ) [static]
```

Sends a notify data buffer for a given connection in the network peer group.

Parameters

<i>g</i>	Pointer to the network peer group.
<i>c</i>	Pointer to the connection.
<i>b</i>	Pointer to the bit buffer containing the data to be sent.
<i>userData</i>	Pointer to user data associated with the buffer.

Returns

True if the buffer was successfully sent, otherwise false.

6.347.2.12 SendReliable()

```
static void SendReliable (  
    NetPeerGroup * g,  
    NetConnection * c,  
    ReliableId rid,  
    byte * data,  
    int dataLength ) [static]
```

Sends reliable data for a given connection in the network peer group.

Parameters

<i>g</i>	Pointer to the network peer group.
<i>c</i>	Pointer to the connection.
<i>rid</i>	Reliable ID for the data being sent.
<i>data</i>	Pointer to the data to be sent.
<i>dataLength</i>	The length of the data to be sent.

6.347.2.13 SendUnconnectedData()

```
static bool SendUnconnectedData (
    NetPeerGroup * g,
    NetAddress address,
    void * data,
    int dataLength ) [static]
```

Sends unconnected data to a specified address.

Parameters

<i>g</i>	Pointer to the network peer group.
<i>address</i>	The address to send the data to.
<i>data</i>	Pointer to the data to be sent.
<i>dataLength</i>	The length of the data to be sent.

Returns

True if the data was successfully sent, otherwise false.

6.347.2.14 SendUnreliableDataBuffer()

```
static bool SendUnreliableDataBuffer (
    NetPeerGroup * g,
    NetConnection * c,
    NetBitBuffer * b ) [static]
```

Sends an unreliable data buffer for a given connection in the network peer group.

Parameters

<i>g</i>	Pointer to the network peer group.
<i>c</i>	Pointer to the connection.
<i>b</i>	Pointer to the bit buffer containing the data to be sent.

Returns

True if the buffer was successfully sent, otherwise false.

6.347.2.15 TryGetConnectionByIndex()

```
static bool TryGetConnectionByIndex (
    NetPeerGroup * g,
    int index,
    out NetConnection * connection ) [static]
```

Tries to get a connection by its index in the network peer group.

Parameters

<i>g</i>	Pointer to the network peer group.
<i>index</i>	The index of the connection to retrieve.
<i>connection</i>	Output parameter that will point to the connection if found.

Returns

True if the connection was found, otherwise false.

6.347.2.16 Update()

```
static void Update (  
    NetPeerGroup * g,  
    INetPeerGroupCallbacks cb ) [static]
```

Updates the network peer group by processing received data, handling timeouts, and managing connection retries.

Parameters

<i>g</i>	Pointer to the network peer group.
<i>cb</i>	Callback interface for network peer group events.

6.347.3 Property Documentation**6.347.3.1 ConnectionCount**

```
int ConnectionCount [get]
```

Gets the number of active connections.

6.347.3.2 Group

```
int Group [get]
```

Gets the group index.

6.347.3.3 Time

```
double Time [get]
```

Gets the elapsed time in seconds.

6.348 NetSendEnvelope Struct Reference

Represents an envelope for sending network packets in the [Fusion.Sockets](#) namespace.

Public Attributes

- double [SendTime](#)
Gets or sets the time the packet was sent.
- ushort [Sequence](#)
Gets or sets the sequence number of the packet.
- void * [UserData](#)
Gets or sets the user data associated with the envelope.

6.348.1 Detailed Description

Represents an envelope for sending network packets in the [Fusion.Sockets](#) namespace.

6.348.2 Member Data Documentation

6.348.2.1 SendTime

```
double SendTime
```

Gets or sets the time the packet was sent.

6.348.2.2 Sequence

```
ushort Sequence
```

Gets or sets the sequence number of the packet.

6.348.2.3 UserData

`void* UserData`

Gets or sets the user data associated with the envelope.

6.349 NetSocket Struct Reference

Represents a network socket with a handle and a native socket.

Public Attributes

- long [Handle](#)
The handle of the socket.
- NanoSockets.Socket [NativeSocket](#)
The native socket.

Properties

- bool [IsCreated](#) [get]
Gets a value indicating whether the socket is created.

6.349.1 Detailed Description

Represents a network socket with a handle and a native socket.

6.349.2 Member Data Documentation

6.349.2.1 Handle

`long Handle`

The handle of the socket.

6.349.2.2 NativeSocket

`NanoSockets.Socket NativeSocket`

The native socket.

6.349.3 Property Documentation

6.349.3.1 IsCreated

```
bool IsCreated [get]
```

Gets a value indicating whether the socket is created.

6.350 ReliableHeader Struct Reference

Represents a reliable header structure used in the [Fusion.Sockets](#) namespace.

Static Public Member Functions

- static byte * [GetData](#) ([ReliableHeader](#) *header)
Gets the data associated with the specified [ReliableHeader](#).

Public Attributes

- [ReliableId](#) Id
The [ReliableId](#) associated with this [ReliableHeader](#).
- [ReliableHeader](#) * [Next](#)
Pointer to the next [ReliableHeader](#) in the list.
- [ReliableHeader](#) * [Prev](#)
Pointer to the previous [ReliableHeader](#) in the list.

Static Public Attributes

- const int [SIZE](#) = 16 + [ReliableId](#).SIZE
The size of the [ReliableHeader](#) structure in bytes.

6.350.1 Detailed Description

Represents a reliable header structure used in the [Fusion.Sockets](#) namespace.

6.350.2 Member Function Documentation

6.350.2.1 GetData()

```
static byte* GetData (  
    ReliableHeader * header ) [static]
```

Gets the data associated with the specified [ReliableHeader](#).

Parameters

<code>header</code>	The ReliableHeader to get data from.
---------------------	--

Returns

A pointer to the data associated with the specified [ReliableHeader](#).

6.350.3 Member Data Documentation

6.350.3.1 Id

`ReliableId` Id

The [ReliableId](#) associated with this [ReliableHeader](#).

6.350.3.2 Next

`ReliableHeader*` Next

Pointer to the next [ReliableHeader](#) in the list.

6.350.3.3 Prev

`ReliableHeader*` Prev

Pointer to the previous [ReliableHeader](#) in the list.

6.350.3.4 SIZE

```
const int SIZE = 16 + ReliableId.SIZE [static]
```

The size of the [ReliableHeader](#) structure in bytes.

6.351 ReliableId Struct Reference

Represents a reliable identifier used in the [Fusion.Sockets](#) namespace.

Public Attributes

- int [_padding](#)
Padding to align the structure.
- [ReliableKey](#) Key
The reliable key associated with this [ReliableId](#).
- ulong [Sequence](#)
The sequence number associated with this [ReliableId](#).
- int [SliceLength](#)
The length of the slice.
- int [Source](#)
The source identifier.
- int [SourceSend](#)
The source send identifier.
- int [Target](#)
The target identifier.
- int [TotalLength](#)
The total length.

Static Public Attributes

- const int [SIZE](#) = 48
The size of the [ReliableId](#) structure in bytes.

Properties

- long [SourceCombined](#) [get]
Gets the combined source identifier.

6.351.1 Detailed Description

Represents a reliable identifier used in the [Fusion.Sockets](#) namespace.

6.351.2 Member Data Documentation

6.351.2.1 [_padding](#)

```
int _padding
```

Padding to align the structure.

6.351.2.2 Key

`ReliableKey` Key

The reliable key associated with this [ReliableId](#).

6.351.2.3 Sequence

`ulong` Sequence

The sequence number associated with this [ReliableId](#).

6.351.2.4 SIZE

```
const int SIZE = 48 [static]
```

The size of the [ReliableId](#) structure in bytes.

6.351.2.5 SliceLength

`int` SliceLength

The length of the slice.

6.351.2.6 Source

`int` Source

The source identifier.

6.351.2.7 SourceSend

`int` SourceSend

The source send identifier.

6.351.2.8 Target

`int Target`

The target identifier.

6.351.2.9 TotalLength

`int TotalLength`

The total length.

6.351.3 Property Documentation

6.351.3.1 SourceCombined

`long SourceCombined [get]`

Gets the combined source identifier.

6.352 ReliableKey Struct Reference

Represents a reliable key structure used in the [Fusion.Sockets](#) namespace.

Public Member Functions

- void [GetInts](#) (out int key0, out int key1, out int key2, out int key3)
Gets the key data as integers.
- void [GetULongs](#) (out ulong key0, out ulong key1)
Gets the key data as unsigned long integers.

Static Public Member Functions

- static [ReliableKey FromInts](#) (int key0=0, int key1=0, int key2=0, int key3=0)
Creates a [ReliableKey](#) from integer values.
- static [ReliableKey FromULongs](#) (ulong key0=0, ulong key1=0)
Creates a [ReliableKey](#) from unsigned long values.

Public Attributes

- fixed byte [Data](#) [16]
Fixed byte array to store the key data.

Static Public Attributes

- const int [SIZE](#) = 16

The size of the [ReliableKey](#) structure in bytes.

6.352.1 Detailed Description

Represents a reliable key structure used in the [Fusion.Sockets](#) namespace.

6.352.2 Member Function Documentation

6.352.2.1 FromInts()

```
static ReliableKey FromInts (  
    int key0 = 0,  
    int key1 = 0,  
    int key2 = 0,  
    int key3 = 0 ) [static]
```

Creates a [ReliableKey](#) from integer values.

Parameters

<i>key0</i>	The first integer key.
<i>key1</i>	The second integer key.
<i>key2</i>	The third integer key.
<i>key3</i>	The fourth integer key.

Returns

A new [ReliableKey](#) instance.

6.352.2.2 FromULongs()

```
static ReliableKey FromULongs (  
    ulong key0 = 0,  
    ulong key1 = 0 ) [static]
```

Creates a [ReliableKey](#) from unsigned long values.

Parameters

<i>key0</i>	The first unsigned long key.
<i>key1</i>	The second unsigned long key.

Returns

A new [ReliableKey](#) instance.

6.352.2.3 GetInts()

```
void GetInts (
    out int key0,
    out int key1,
    out int key2,
    out int key3 )
```

Gets the key data as integers.

Parameters

<i>key0</i>	The first integer key.
<i>key1</i>	The second integer key.
<i>key2</i>	The third integer key.
<i>key3</i>	The fourth integer key.

6.352.2.4 GetUlongs()

```
void GetUlongs (
    out ulong key0,
    out ulong key1 )
```

Gets the key data as unsigned long integers.

Parameters

<i>key0</i>	The first unsigned long key.
<i>key1</i>	The second unsigned long key.

6.352.3 Member Data Documentation**6.352.3.1 Data**

```
fixed byte Data[16]
```

Fixed byte array to store the key data.

6.352.3.2 SIZE

```
const int SIZE = 16 [static]
```

The size of the [ReliableKey](#) structure in bytes.

6.353 ReliableList Struct Reference

Represents a list of reliable headers.

Public Member Functions

- void [AddAfter](#) ([ReliableHeader](#) *after, [ReliableHeader](#) *item)
Adds the specified item after another specified item in the list.
- void [AddBefore](#) ([ReliableHeader](#) *before, [ReliableHeader](#) *item)
Adds the specified item before another specified item in the list.
- void [AddFirst](#) ([ReliableHeader](#) *item)
Adds the specified item to the beginning of the list.
- void [AddLast](#) ([ReliableHeader](#) *item)
Adds the specified item to the end of the list.
- bool [IsInList](#) ([ReliableHeader](#) *item)
- void [Remove](#) ([ReliableHeader](#) *item)
Removes the specified item from the list.
- [ReliableHeader](#) * [RemoveHead](#) ()
Removes and returns the head element of the list.

Public Attributes

- int [Count](#)
Gets or sets the number of items in the list.
- [ReliableHeader](#) * [Head](#)
Gets or sets the head of the list.
- [ReliableHeader](#) * [Tail](#)
Gets or sets the tail of the list.

6.353.1 Detailed Description

Represents a list of reliable headers.

6.353.2 Member Function Documentation

6.353.2.1 AddAfter()

```
void AddAfter (  
    ReliableHeader * after,  
    ReliableHeader * item )
```

Adds the specified item after another specified item in the list.

Parameters

<i>after</i>	The item after which the new item will be added.
<i>item</i>	The item to add to the list.

6.353.2.2 AddBefore()

```
void AddBefore (
    ReliableHeader * before,
    ReliableHeader * item )
```

Adds the specified item before another specified item in the list.

Parameters

<i>before</i>	The item before which the new item will be added.
<i>item</i>	The item to add to the list.

6.353.2.3 AddFirst()

```
void AddFirst (
    ReliableHeader * item )
```

Adds the specified item to the beginning of the list.

Parameters

<i>item</i>	The item to add to the beginning of the list.
-------------	---

6.353.2.4 AddLast()

```
void AddLast (
    ReliableHeader * item )
```

Adds the specified item to the end of the list.

Parameters

<i>item</i>	The item to add to the end of the list.
-------------	---

6.353.2.5 Remove()

```
void Remove (
    ReliableHeader * item )
```

Removes the specified item from the list.

Parameters

<i>item</i>	The item to remove.
-------------	---------------------

6.353.2.6 RemoveHead()

```
ReliableHeader* RemoveHead ( )
```

Removes and returns the head element of the list.

Returns

The head element of the list.

6.353.3 Member Data Documentation

6.353.3.1 Count

```
int Count
```

Gets or sets the number of items in the list.

6.353.3.2 Head

```
ReliableHeader* Head
```

Gets or sets the head of the list.

6.353.3.3 Tail

```
ReliableHeader* Tail
```

Gets or sets the tail of the list.

6.354 StunServers.StunServer Class Reference

Stores Addresses of a STUN Server

Public Member Functions

- override string **ToString** ()

Public Attributes

- [NetAddress](#) **IPv4Addr**
- [NetAddress](#) **IPv6Addr**

Properties

- bool **HasIPv4Support** [get]
- bool **HasIPv6Support** [get]
- static [IEqualityComparer](#)< [StunServer](#) > **StunServerEqualityComparer** = new [Pv4AddrEquality](#)←
Comparer() [get]

6.354.1 Detailed Description

Stores Addresses of a STUN Server

6.355 StartGameArgs Struct Reference

[Fusion](#) Start Arguments, used to configure the simulation mode and other settings

Public Member Functions

- override string [ToString](#) ()
StartGameArgs ToString()

Public Attributes

- [NetAddress?](#) [Address](#)
Peer Binding Address
- [AuthenticationValues](#) [AuthValues](#)
Custom Authentication Data
- [NetworkProjectConfig](#) [Config](#)
Custom [NetworkProjectConfig](#) used to start the simulation
- [byte\[\]](#) [ConnectionToken](#)
Connection token sent by client to server. Not used in shared mode.
- [Type\[\]](#) [CustomCallbackInterfaces](#)
User defined callback interfaces we will provide O(1) constant time lookup for
- [string](#) [CustomLobbyName](#)
Session Custom Lobby to be published in
- [FusionAppSettings](#) [CustomPhotonAppSettings](#)
Custom Photon Application Settings
- [NetAddress?](#) [CustomPublicAddress](#)
Custom Public Reflexive Address
- [string](#) [CustomSTUNServer](#)
Specify a Custom STUN Server used to Resolve the peer Reflexive Addresses It can be used to specify multiple STUN Servers separated by semicolon (;)
- [bool](#) [DisableNATPunchthrough](#)
Flag to disable the NAT Punchthrough implementation and connect only via Relay
- [bool?](#) [EnableClientSessionCreation](#)
Enables the Session creation when starting a Client with an specific Session Name
- [GameMode](#) [GameMode](#)
[Fusion.GameMode](#) in which this peer will start
- [Action<](#) [NetworkRunner](#) [>](#) [HostMigrationResume](#)
Callback invoked when the new Host is migrating from the old Host state
- [HostMigrationToken](#) [HostMigrationToken](#)
Host Migration Token used when restarting the [Fusion Simulation](#)
- [bool?](#) [IsOpen](#)
Session should be created Open or Closed to accept joins
- [bool?](#) [IsVisible](#)
Session should be Visible or not in the Session Lobby list
- [MatchmakingMode?](#) [MatchmakingMode](#)
*Session Join Matchmaking Mode when joining a Session. For more information, check [Fusion.Photon.Realtime](#).↔
[MatchmakingMode](#)*
- [INetworkObjectInitializer](#) [ObjectInitializer](#)
See [INetworkRunnerUpdater](#)
- [INetworkObjectProvider](#) [ObjectProvider](#)
Object pool to use
- [Action<](#) [NetworkRunner](#) [>](#) [OnGameStarted](#)
Callback that is invoked when the [Fusion](#) has fully started
- [int?](#) [PlayerCount](#)
Number of players allowed to connect to the session.
- [NetworkSceneInfo?](#) [Scene](#)
Scene that will be set as the starting Scene.
- [INetworkSceneManager](#) [SceneManager](#)
See [INetworkSceneManager](#).
- [string](#) [SessionName](#)

Photon Cloud Session Name used either to Create or Join a Session.

- Func< string > [SessionNameGenerator](#)

Used to generate a new Session Name when creating a Session.

- Dictionary< string, SessionProperty > [SessionProperties](#)

Custom Session Properties. This dictionary can be used to either setup the initial Session Properties when creating a Session but also to set the matchmaking filters when joining a Random Session.

- CancellationToken [StartGameCancellationToken](#)

Optional CancellationToken used to cancel the [NetworkRunner](#) start up process and shutdown

- [INetworkRunnerUpdater Updater](#)

See [INetworkRunnerUpdater](#)

- bool? [UseCachedRegions](#)

Enables the usage of the previous cached regions pings. This speeds up the region ping process and the runner startup process.

- bool? [UseDefaultPhotonCloudPorts](#)

Signal if the internal Realtime Client should use the Default Photon ports to connect to the Photon Cloud. By default, [Fusion](#) uses ports: 27000, 27001 and 27002. Set this to True to use ports: 5058, 5055 and 5056.

6.355.1 Detailed Description

[Fusion](#) Start Arguments, used to configure the simulation mode and other settings

More about matchmaking: <https://doc.photonengine.com/en-us/fusion/current/manual/matchmaking>

6.355.2 Member Function Documentation

6.355.2.1 ToString()

```
override string ToString ( )
```

[StartGameArgs ToString\(\)](#)

6.355.3 Member Data Documentation

6.355.3.1 Address

[NetAddress?](#) Address

Peer Binding Address

Default: [NetAddress.Any\(ushort\)](#)

6.355.3.2 AuthValues

AuthenticationValues AuthValues

Custom Authentication Data

Default: null (default authentication values)

6.355.3.3 Config

NetworkProjectConfig Config

Custom [NetworkProjectConfig](#) used to start the simulation

Default: Global [NetworkProjectConfig](#)

More about [NetworkProjectConfig](#): <https://doc.photonengine.com/en-us/fusion/current/manual/network>

6.355.3.4 ConnectionToken

byte [] ConnectionToken

Connection token sent by client to server. Not used in shared mode.

Default: null (empty connection token)

6.355.3.5 CustomCallbackInterfaces

Type [] CustomCallbackInterfaces

User defined callback interfaces we will provide O(1) constant time lookup for

Default: null

6.355.3.6 CustomLobbyName

string CustomLobbyName

Session Custom Lobby to be published in

Default: null (default Lobby for each Session Type, LobbyClientServer or LobbyShared)

6.355.3.7 CustomPhotonAppSettings

FusionAppSettings CustomPhotonAppSettings

Custom Photon Application Settings

Default: null (Global PhotonAppSettings)

6.355.3.8 CustomPublicAddress

[NetAddress?](#) CustomPublicAddress

Custom Public Reflexive Address

Default: null

6.355.3.9 CustomSTUNServer

`string CustomSTUNServer`

Specify a Custom STUN Server used to Resolve the peer Reflexive Addresses It can be used to specify multiple STUN Servers separated by semicolon (;)

Default: null (no custom STUN Server)

6.355.3.10 DisableNATPunchthrough

`bool DisableNATPunchthrough`

Flag to disable the NAT Punchthrough implementation and connect only via Relay

Default: false

6.355.3.11 EnableClientSessionCreation

`bool? EnableClientSessionCreation`

Enables the Session creation when starting a Client with an specific Session Name

Default: false (clients *can not* create new Sessions)

6.355.3.12 GameMode

`GameMode GameMode`

[Fusion.GameMode](#) in which this peer will start

6.355.3.13 HostMigrationResume

`Action<NetworkRunner> HostMigrationResume`

Callback invoked when the new Host is migrating from the old Host state

Default: null

6.355.3.14 HostMigrationToken

[HostMigrationToken](#) `HostMigrationToken`

Host Migration Token used when restarting the [Fusion Simulation](#)

Default: null

6.355.3.15 IsOpen

`bool? IsOpen`

Session should be created Open or Closed to accept joins

Default: true

6.355.3.16 IsVisible

`bool? IsVisible`

Session should be Visible or not in the Session Lobby list

Default: true

6.355.3.17 MatchmakingMode

`MatchmakingMode? MatchmakingMode`

Session Join Matchmaking Mode when joining a Session. For more information, check [Fusion.Photon.Realtime.MatchmakingMode](#)

Default: `MatchmakingMode.FillRoom`

6.355.3.18 ObjectInitializer

[INetworkObjectInitializer](#) `ObjectInitializer`

See [INetworkRunnerUpdater](#)

6.355.3.19 ObjectProvider

[INetworkObjectProvider](#) `ObjectProvider`

Object pool to use

Default: null

6.355.3.20 OnGameStarted

Action<NetworkRunner> OnGameStarted

Callback that is invoked when the Fusion has fully started

Default: null

6.355.3.21 PlayerCount

int? PlayerCount

Number of players allowed to connect to the session.

Default: DefaultPlayers from the Global NetworkProjectConfig

6.355.3.22 Scene

NetworkSceneInfo? Scene

Scene that will be set as the starting Scene.

Default: null (no scene set)

6.355.3.23 SceneManager

INetworkSceneManager SceneManager

See INetworkSceneManager.

Default: null

More about Scene Loading: <https://doc.photonengine.com/en-us/fusion/current/manual/scene-load>

6.355.3.24 SessionName

string SessionName

Photon Cloud Session Name used either to Create or Join a Session.

Default: null (random session matching)

6.355.3.25 SessionNameGenerator

Func<string> SessionNameGenerator

Used to generate a new Session Name when creating a Session.

Default: null (a random session name is generated based on a GUID)

6.355.3.26 SessionProperties

Dictionary<string, SessionProperty> SessionProperties

Custom Session Properties. This dictionary can be used to either setup the initial Session Properties when creating a Session but also to set the matchmaking filters when joining a Random Session.

Default: null (empty custom properties)

6.355.3.27 StartGameCancellationToken

CancellationToken StartGameCancellationToken

Optional CancellationToken used to cancel the [NetworkRunner](#) start up process and shutdown

Defaults: null

6.355.3.28 Updater

[INetworkRunnerUpdater](#) Updater

See [INetworkRunnerUpdater](#)

6.355.3.29 UseCachedRegions

bool? UseCachedRegions

Enables the usage of the previous cached regions pings. This speeds up the region ping process and the runner startup process.

Defaults: true

6.355.3.30 UseDefaultPhotonCloudPorts

bool? UseDefaultPhotonCloudPorts

Signal if the internal Realtime Client should use the Default Photon ports to connect to the Photon Cloud. By default, [Fusion](#) uses ports: 27000, 27001 and 27002. Set this to True to use ports: 5058, 5055 and 5056.

Default: false (uses ports 27000, 27001 and 27002)

6.356 StartGameResult Class Reference

Represents the result of starting the [Fusion Simulation](#)

Public Member Functions

- override string [ToString](#) ()
String representation of the [StartGameResult](#)

Properties

- string [ErrorMessage](#) [get]
Custom Error Message filled with data about the Shutdown. Usually used to store custom data when the StartGame fails.
- bool [Ok](#) [get]
Signal if the Start was OK
- [ShutdownReason ShutdownReason](#) [get]
Start Game Shutdown Reason
- string [StackTrace](#) [get]
Optional Exception StackTrace

6.356.1 Detailed Description

Represents the result of starting the [Fusion Simulation](#)

6.356.2 Member Function Documentation

6.356.2.1 ToString()

```
override string ToString ( )
```

String representation of the [StartGameResult](#)

6.356.3 Property Documentation

6.356.3.1 ErrorMessage

```
string ErrorMessage [get]
```

Custom Error Message filled with data about the Shutdown. Usually used to store custom data when the StartGame fails.

6.356.3.2 Ok

`bool Ok [get]`

Signal if the Start was OK

6.356.3.3 ShutdownReason

`ShutdownReason ShutdownReason [get]`

Start Game Shutdown Reason

6.356.3.4 StackTrace

`string StackTrace [get]`

Optional Exception StackTrace

6.357 BehaviourStatisticsManager Class Reference

[Behaviour](#) statistics manager will provide access to the behaviour statistics snapshots.

Properties

- [BehaviourStatisticsSnapshot CompletedSnapshot](#) [get]
The completed snapshot of statistics related to [Fusion Behaviour](#) type execution from the previous update.

6.357.1 Detailed Description

[Behaviour](#) statistics manager will provide access to the behaviour statistics snapshots.

6.357.2 Property Documentation

6.357.2.1 CompletedSnapshot

`BehaviourStatisticsSnapshot CompletedSnapshot [get]`

The completed snapshot of statistics related to [Fusion Behaviour](#) type execution from the previous update.

6.358 BehaviourStatisticsSnapshot Class Reference

Represents a snapshot of statistics related to [Fusion Behaviour](#) type execution.

Properties

- int [FixedUpdateNetworkExecutionCount](#) [get]
Gets the count of FixedUpdateNetwork executions.
- double [FixedUpdateNetworkExecutionTime](#) [get]
Gets the total execution time of FixedUpdateNetwork in milliseconds.
- int [RenderExecutionCount](#) [get]
Gets the count of Render executions.
- double [RenderExecutionTime](#) [get]
Gets the total execution time of Render in milliseconds.

6.358.1 Detailed Description

Represents a snapshot of statistics related to [Fusion Behaviour](#) type execution.

6.358.2 Property Documentation

6.358.2.1 FixedUpdateNetworkExecutionCount

```
int FixedUpdateNetworkExecutionCount [get]
```

Gets the count of FixedUpdateNetwork executions.

6.358.2.2 FixedUpdateNetworkExecutionTime

```
double FixedUpdateNetworkExecutionTime [get]
```

Gets the total execution time of FixedUpdateNetwork in milliseconds.

6.358.2.3 RenderExecutionCount

```
int RenderExecutionCount [get]
```

Gets the count of Render executions.

6.358.2.4 RenderExecutionTime

`double RenderExecutionTime [get]`

Gets the total execution time of Render in milliseconds.

6.359 FusionStatisticsManager Class Reference

Represents a fusion statistics manager.

Properties

- [FusionStatisticsSnapshot CompleteSnapshot](#) [get]
Provides access to a complete snapshot of the fusion statistics.
- [NetworkObjectStatisticsManager ObjectStatisticsManager](#) [get]
Manages the network object statistics.

6.359.1 Detailed Description

Represents a fusion statistics manager.

6.359.2 Property Documentation

6.359.2.1 CompleteSnapshot

`FusionStatisticsSnapshot CompleteSnapshot [get]`

Provides access to a complete snapshot of the fusion statistics.

This property behaves as a read-only property and provides the collected snapshot of the fusion statistics.

6.359.2.2 ObjectStatisticsManager

`NetworkObjectStatisticsManager ObjectStatisticsManager [get]`

Manages the network object statistics.

6.360 FusionStatisticsSnapshot Class Reference

Represents a snapshot of [Fusion](#) statistics.

Properties

- int [ForwardTicks](#) [get]
Gets or sets the number of forward ticks.
- int [GeneralAllocMemoryFreeInBytes](#) [get]
The number of free segments allocated for general purposes on the simulation.
- int [GeneralAllocMemoryUsedInBytes](#) [get]
The number of used segments allocated for general purposes on the simulation.
- float [InBandwidth](#) [get]
Gets or sets the in-bandwidth value.
- int [InObjectUpdates](#) [get]
The number of received objects updates per packet.
- int [InPackets](#) [get]
Gets or sets the number of incoming packets.
- float [InputInBandwidth](#) [get]
Gets the input in bandwidth.
- float [InputOutBandwidth](#) [get]
Gets the input out bandwidth.
- float [InputReceiveDelta](#) [get]
Gets the Input Receive Delta.
- float [InterpolationOffset](#) [get]
Gets the Interpolation Offset.
- float [InterpolationSpeed](#) [get]
Gets the Interpolation Speed.
- int [ObjectsAllocMemoryFreeInBytes](#) [get]
The number of free segments allocated for objects on the simulation.
- int [ObjectsAllocMemoryUsedInBytes](#) [get]
The number of used segments allocated for objects on the simulation.
- float [OutBandwidth](#) [get]
Gets or sets the outbandwith property.
- int [OutObjectUpdates](#) [get]
The number of sent objects updates per packet.
- int [OutPackets](#) [get]
Gets or sets the number of outgoing packets.
- int [Resimulations](#) [get]
Gets or sets the number of resimulations.
- float [RoundTripTime](#) [get]
Gets or sets the round trip time (RTT) in seconds.
- float [SimulationSpeed](#) [get]
*Gets the *Simulation* Speed.*
- float [SimulationTimeOffset](#) [get]
*Gets the *Simulation* Time Offset.*
- float [StateReceiveDelta](#) [get]
Gets the State Receive Delta.
- int [TimeResets](#) [get]
Gets the Time Resets count.
- int [WordsReadCount](#) [get]
The number of words that were read.
- int [WordsReadSize](#) [get]
The total size in bytes of all read words.
- int [WordsWrittenCount](#) [get]
The number of words that were written.
- int [WordsWrittenSize](#) [get]
The total size in bytes of all written words.

6.360.1 Detailed Description

Represents a snapshot of [Fusion](#) statistics.

6.360.2 Property Documentation

6.360.2.1 ForwardTicks

```
int ForwardTicks [get]
```

Gets or sets the number of forward ticks.

6.360.2.2 GeneralAllocMemoryFreeInBytes

```
int GeneralAllocMemoryFreeInBytes [get]
```

The number of free segments allocated for general purposes on the simulation.

6.360.2.3 GeneralAllocMemoryUsedInBytes

```
int GeneralAllocMemoryUsedInBytes [get]
```

The number of used segments allocated for general purposes on the simulation.

6.360.2.4 InBandwidth

```
float InBandwidth [get]
```

Gets or sets the in-bandwidth value.

6.360.2.5 InObjectUpdates

```
int InObjectUpdates [get]
```

The number of received objects updates per packet.

6.360.2.6 InPackets

`int InPackets [get]`

Gets or sets the number of incoming packets.

6.360.2.7 InputInBandwidth

`float InputInBandwidth [get]`

Gets the input in bandwidth.

6.360.2.8 InputOutBandwidth

`float InputOutBandwidth [get]`

Gets the input out bandwidth.

6.360.2.9 InputReceiveDelta

`float InputReceiveDelta [get]`

Gets the Input Receive Delta.

6.360.2.10 InterpolationOffset

`float InterpolationOffset [get]`

Gets the Interpolation Offset.

6.360.2.11 InterpolationSpeed

`float InterpolationSpeed [get]`

Gets the Interpolation Speed.

6.360.2.12 ObjectsAllocMemoryFreeInBytes

```
int ObjectsAllocMemoryFreeInBytes [get]
```

The number of free segments allocated for objects on the simulation.

6.360.2.13 ObjectsAllocMemoryUsedInBytes

```
int ObjectsAllocMemoryUsedInBytes [get]
```

The number of used segments allocated for objects on the simulation.

6.360.2.14 OutBandwidth

```
float OutBandwidth [get]
```

Gets or sets the outbandwidth property.

6.360.2.15 OutObjectUpdates

```
int OutObjectUpdates [get]
```

The number of sent objects updates per packet.

6.360.2.16 OutPackets

```
int OutPackets [get]
```

Gets or sets the number of outgoing packets.

6.360.2.17 Resimulations

```
int Resimulations [get]
```

Gets or sets the number of resimulations.

6.360.2.18 RoundTripTime

```
float RoundTripTime [get]
```

Gets or sets the round trip time (RTT) in seconds.

6.360.2.19 SimulationSpeed

```
float SimulationSpeed [get]
```

Gets the [Simulation](#) Speed.

6.360.2.20 SimulationTimeOffset

```
float SimulationTimeOffset [get]
```

Gets the [Simulation](#) Time Offset.

6.360.2.21 StateReceiveDelta

```
float StateReceiveDelta [get]
```

Gets the State Receive Delta.

6.360.2.22 TimeResets

```
int TimeResets [get]
```

Gets the Time Resets count.

6.360.2.23 WordsReadCount

```
int WordsReadCount [get]
```

The number of words that were read.

6.360.2.24 WordsReadSize

`int WordsReadSize [get]`

The total size in bytes of all read words.

6.360.2.25 WordsWrittenCount

`int WordsWrittenCount [get]`

The number of words that were written.

6.360.2.26 WordsWrittenSize

`int WordsWrittenSize [get]`

The total size in bytes of all written words.

6.361 LagCompensationStatisticsSnapshot Class Reference

Represents a snapshot of lag compensation statistics.

Properties

- double [AddOnBufferTime](#) [get]
Represents the amount of time taken to register new hitboxes on the HitboxBuffer.
- double [AddOnBVHTime](#) [get]
Represents the amount of time taken to register new hitboxes on the BVH.
- double [AdvanceBufferTime](#) [get]
Represents the amount of time taken to advance the hitbox buffer and copy previous snapshot information.
- int [BVHMaxDeep](#) [get]
Represents the deeper level reached by the BVH.
- int [BVHNodesCount](#) [get]
Represents total nodes count on the BVH.
- int [HitboxesCount](#) [get]
Represents the total number of HitBoxCollider on the HitBoxSnapshot colliders buffer, including both used and free ones.
- double [RefitBVHTime](#) [get]
Represents the amount of time taken to refit the BVH bounds after the nodes update.
- double [TotalElapsedTime](#) [get]
Represents the total time taken to advance, add and update all hitboxes on the Lag Compensation system.
- double [UpdateBufferTime](#) [get]
Represents the amount of time taken to update the HitboxBuffer.
- double [UpdateBVHTime](#) [get]
Represents the amount of time taken to update the BVH.

6.361.1 Detailed Description

Represents a snapshot of lag compensation statistics.

6.361.2 Property Documentation

6.361.2.1 AddOnBufferTime

```
double AddOnBufferTime [get]
```

Represents the amount of time taken to register new hitboxes on the HitboxBuffer.

6.361.2.2 AddOnBVHTime

```
double AddOnBVHTime [get]
```

Represents the amount of time taken to register new hitboxes on the BVH.

6.361.2.3 AdvanceBufferTime

```
double AdvanceBufferTime [get]
```

Represents the amount of time taken to advance the hitbox buffer and copy previous snapshot information.

6.361.2.4 BVHMaxDeep

```
int BVHMaxDeep [get]
```

Represents the deeper level reached by the BVH.

6.361.2.5 BVHNodesCount

```
int BVHNodesCount [get]
```

Represents total nodes count on the BVH.

6.361.2.6 HitboxesCount

```
int HitboxesCount [get]
```

Represents the total number of HitBoxCollider on the HitBoxSnapshot colliders buffer, including both used and free ones.

6.361.2.7 RefitBVHTime

```
double RefitBVHTime [get]
```

Represents the amount of time taken to refit the BVH bounds after the nodes update.

6.361.2.8 TotalElapsedTime

```
double TotalElapsedTime [get]
```

Represents the total time taken to advance, add and update all hitboxes on the Lag Compensation system.

6.361.2.9 UpdateBufferTime

```
double UpdateBufferTime [get]
```

Represents the amount of time taken to update the HitboxBuffer.

6.361.2.10 UpdateBVHTime

```
double UpdateBVHTime [get]
```

Represents the amount of time taken to update the BVH.

6.362 MemoryStatisticsSnapshot Struct Reference

Represents a snapshot of specific memory statistics. For basic total allocated memory and total free memory check the [Fusion Statistics](#).

Public Types

- enum class [TargetAllocator](#)
Enum representing the target allocator for memory statistics.

Public Attributes

- `int[]` [BucketFreeBlocksCount](#)
Represents the number of free blocks in each bucket of the allocator.
- `int[]` [BucketFullBlocksCount](#)
Represents the number of full blocks in each bucket of the allocator.
- `int[]` [BucketUsedBlocksCount](#)
Represents the number of used blocks in each bucket of the allocator.
- `int` [TotalFreeBlocks](#)
Represents the total number of free blocks in the allocator.

Static Public Attributes

- `const int` [BUCKET_COUNT](#) = `Allocator.BUCKET_COUNT`
The number of buckets used in the allocator.

6.362.1 Detailed Description

Represents a snapshot of specific memory statistics. For basic total allocated memory and total free memory check the [Fusion Statistics](#).

6.362.2 Member Enumeration Documentation

6.362.2.1 TargetAllocator

```
enum TargetAllocator [strong]
```

Enum representing the target allocator for memory statistics.

6.362.3 Member Data Documentation

6.362.3.1 BUCKET_COUNT

```
const int BUCKET\_COUNT = Allocator.BUCKET_COUNT [static]
```

The number of buckets used in the allocator.

6.362.3.2 BucketFreeBlocksCount

```
int [] BucketFreeBlocksCount
```

Represents the number of free blocks in each bucket of the allocator.

6.362.3.3 BucketFullBlocksCount

```
int [] BucketFullBlocksCount
```

Represents the number of full blocks in each bucket of the allocator.

6.362.3.4 BucketUsedBlocksCount

```
int [] BucketUsedBlocksCount
```

Represents the number of used blocks in each bucket of the allocator.

6.362.3.5 TotalFreeBlocks

```
int TotalFreeBlocks
```

Represents the total number of free blocks in the allocator.

6.363 NetworkObjectStatisticsManager Class Reference

Manages network object statistics for monitored network objects.

Public Member Functions

- void [ClearMonitoredNetworkObjects](#) ()
Clears the list of monitored network objects.
- bool [GetNetworkObjectStatistics](#) ([NetworkId](#) id, out [NetworkObjectStatisticsSnapshot](#) objectStatistics↔
Snapshot)
Retrieves the network object statistics for a given network ID.
- void [MonitorNetworkObjectStatistics](#) ([NetworkId](#) id, bool monitor)
Starts or stops monitoring the statistics of a network object.

6.363.1 Detailed Description

Manages network object statistics for monitored network objects.

6.363.2 Member Function Documentation

6.363.2.1 ClearMonitoredNetworkObjects()

```
void ClearMonitoredNetworkObjects ( )
```

Clears the list of monitored network objects.

6.363.2.2 GetNetworkObjectStatistics()

```
bool GetNetworkObjectStatistics (
    NetworkId id,
    out NetworkObjectStatisticsSnapshot objectStatisticsSnapshot )
```

Retrieves the network object statistics for a given network ID.

Returns

Returns true if the object is being monitored and its statistics are successfully retrieved; otherwise, returns false.

6.363.2.3 MonitorNetworkObjectStatistics()

```
void MonitorNetworkObjectStatistics (
    NetworkId id,
    bool monitor )
```

Starts or stops monitoring the statistics of a network object.

Parameters

<i>id</i>	The identifier of the network object to monitor.
<i>monitor</i>	True to start monitoring, false to stop monitoring.

6.364 NetworkObjectStatisticsSnapshot Class Reference

Represents a snapshot of network object statistics.

Properties

- float `InBandwidth` [get]
Gets or sets the in-bandwidth value.
- int `InPackets` [get]
Gets or sets the number of incoming packets.
- float `OutBandwidth` [get]
Gets or sets the outbandwith property.
- int `OutPackets` [get]
Gets or sets the number of outgoing packets.

6.364.1 Detailed Description

Represents a snapshot of network object statistics.

6.364.2 Property Documentation

6.364.2.1 InBandwidth

```
float InBandwidth [get]
```

Gets or sets the in-bandwidth value.

6.364.2.2 InPackets

```
int InPackets [get]
```

Gets or sets the number of incoming packets.

6.364.2.3 OutBandwidth

```
float OutBandwidth [get]
```

Gets or sets the outbandwith property.

6.364.2.4 OutPackets

```
int OutPackets [get]
```

Gets or sets the number of outgoing packets.

6.365 Tick Struct Reference

A tick is a 32-bit integer that represents a frame number.

Inherits `Comparable< Tick >`, and `IEquatable< Tick >`.

Classes

- class [EqualityComparer](#)
Provides a mechanism for comparing two `Tick` objects for equality.
- class [RelationalComparer](#)
Provides a mechanism for comparing two `Tick` objects.

Public Member Functions

- int [CompareTo](#) (`Tick` other)
Compares the current `Tick` with another `Tick`.
- override bool [Equals](#) (object obj)
Determines whether the specified object is equal to the current `Tick`.
- bool [Equals](#) (`Tick` other)
Determines whether the specified `Tick` is equal to the current `Tick`.
- override int [GetHashCode](#) ()
Serves as the default hash function.
- [Tick Next](#) (int increment)
Returns the next `Tick`, incremented by a specified value.
- override string [ToString](#) ()
String representation of the `Tick`.

Static Public Member Functions

- static implicit [operator bool](#) (`Tick` value)
Converts a `Tick` to a boolean.
- static implicit [operator int](#) (`Tick` value)
Converts a `Tick` to an integer.
- static implicit [operator Tick](#) (int value)
Converts an integer to a `Tick`.
- static bool [operator!=](#) (`Tick` a, `Tick` b)
Determines whether the first specified `Tick` is not equal to the second specified `Tick`.
- static bool [operator<](#) (`Tick` a, `Tick` b)
Determines whether the first specified `Tick` is less than the second specified `Tick`.
- static bool [operator<=](#) (`Tick` a, `Tick` b)
Determines whether the first specified `Tick` is less than or equal to the second specified `Tick`.
- static bool [operator==](#) (`Tick` a, `Tick` b)
Determines whether the first specified `Tick` is equal to the second specified `Tick`.
- static bool [operator>](#) (`Tick` a, `Tick` b)
Determines whether the first specified `Tick` is greater than the second specified `Tick`.
- static bool [operator>=](#) (`Tick` a, `Tick` b)
Determines whether the first specified `Tick` is greater than or equal to the second specified `Tick`.

Public Attributes

- int [Raw](#)

The raw value of the [Tick](#). This represents a frame number.

Static Public Attributes

- const int [ALIGNMENT](#) = 4

The alignment of the [Tick](#) structure in bytes.

- const int [SIZE](#) = 4

The size of the [Tick](#) structure in bytes.

6.365.1 Detailed Description

A tick is a 32-bit integer that represents a frame number.

6.365.2 Member Function Documentation

6.365.2.1 CompareTo()

```
int CompareTo (  
    Tick other )
```

Compares the current [Tick](#) with another [Tick](#).

Parameters

<i>other</i>	A Tick to compare with this Tick .
--------------	--

Returns

A value that indicates the relative order of the objects being compared.

6.365.2.2 Equals() [1/2]

```
override bool Equals (  
    object obj )
```

Determines whether the specified object is equal to the current [Tick](#).

Parameters

<i>obj</i>	The object to compare with the current Tick .
------------	---

Returns

true if the specified object is equal to the current [Tick](#); otherwise, false.

6.365.2.3 Equals() [2/2]

```
bool Equals (  
    Tick other )
```

Determines whether the specified [Tick](#) is equal to the current [Tick](#).

Parameters

<i>other</i>	The Tick to compare with the current Tick .
--------------	---

Returns

true if the specified [Tick](#) is equal to the current [Tick](#); otherwise, false.

6.365.2.4 GetHashCode()

```
override int GetHashCode ( )
```

Serves as the default hash function.

Returns

A hash code for the current [Tick](#).

6.365.2.5 Next()

```
Tick Next (  
    int increment )
```

Returns the next [Tick](#), incremented by a specified value.

Parameters

<i>increment</i>	The value to increment the Tick by.
------------------	---

Returns

A new [Tick](#) that represents the current [Tick](#) incremented by the specified value.

6.365.2.6 operator bool()

```
static implicit operator bool (  
    Tick value ) [static]
```

Converts a [Tick](#) to a boolean.

Parameters

<i>value</i>	The Tick to convert.
--------------	--------------------------------------

Returns

true if the [Tick](#)'s raw value is greater than 0; otherwise, false.

6.365.2.7 operator int()

```
static implicit operator int (  
    Tick value ) [static]
```

Converts a [Tick](#) to an integer.

Parameters

<i>value</i>	The Tick to convert.
--------------	--------------------------------------

Returns

An integer that represents the raw value of the specified [Tick](#).

6.365.2.8 operator Tick()

```
static implicit operator Tick (  
    int value ) [static]
```

Converts an integer to a [Tick](#).

Parameters

<i>value</i>	The integer to convert.
--------------	-------------------------

Returns

A [Tick](#) that represents the specified integer. If the integer is less than 0, the [Tick](#)'s raw value is set to 0.

6.365.2.9 operator"!=()

```
static bool operator!= (  
    Tick a,  
    Tick b ) [static]
```

Determines whether the first specified [Tick](#) is not equal to the second specified [Tick](#).

6.365.2.10 operator<()

```
static bool operator< (  
    Tick a,  
    Tick b ) [static]
```

Determines whether the first specified [Tick](#) is less than the second specified [Tick](#).

6.365.2.11 operator<=()

```
static bool operator<= (  
    Tick a,  
    Tick b ) [static]
```

Determines whether the first specified [Tick](#) is less than or equal to the second specified [Tick](#).

6.365.2.12 operator==(())

```
static bool operator==( (  
    Tick a,  
    Tick b ) [static]
```

Determines whether the first specified [Tick](#) is equal to the second specified [Tick](#).

6.365.2.13 operator>()

```
static bool operator> (
    Tick a,
    Tick b ) [static]
```

Determines whether the first specified [Tick](#) is greater than the second specified [Tick](#).

6.365.2.14 operator>=()

```
static bool operator>= (
    Tick a,
    Tick b ) [static]
```

Determines whether the first specified [Tick](#) is greater than or equal to the second specified [Tick](#).

6.365.2.15 ToString()

```
override string ToString ( )
```

String representation of the [Tick](#).

6.365.3 Member Data Documentation

6.365.3.1 ALIGNMENT

```
const int ALIGNMENT = 4 [static]
```

The alignment of the [Tick](#) structure in bytes.

6.365.3.2 Raw

```
int Raw
```

The raw value of the [Tick](#). This represents a frame number.

6.365.3.3 SIZE

```
const int SIZE = 4 [static]
```

The size of the [Tick](#) structure in bytes.

6.366 Tick.EqualityComparer Class Reference

Provides a mechanism for comparing two [Tick](#) objects for equality.

Inherits `IEqualityComparer< Tick >`.

Public Member Functions

- `bool Equals (Tick x, Tick y)`
Determines whether the specified [Tick](#) objects are equal.
- `int GetHashCode (Tick obj)`
Serves as a hash function for a [Tick](#) object.

6.366.1 Detailed Description

Provides a mechanism for comparing two [Tick](#) objects for equality.

6.366.2 Member Function Documentation

6.366.2.1 Equals()

```
bool Equals (  
    Tick x,  
    Tick y )
```

Determines whether the specified [Tick](#) objects are equal.

Parameters

<code>x</code>	The first Tick object to compare.
<code>y</code>	The second Tick object to compare.

Returns

true if the specified [Tick](#) objects are equal; otherwise, false.

6.366.2.2 GetHashCode()

```
int GetHashCode (
    Tick obj )
```

Serves as a hash function for a [Tick](#) object.

Parameters

<i>obj</i>	The Tick object for which to get a hash code.
------------	---

Returns

A hash code for the specified [Tick](#) object.

6.367 Tick.RelationalComparer Class Reference

Provides a mechanism for comparing two [Tick](#) objects.

Inherits [IComparer< Tick >](#).

Public Member Functions

- int [Compare](#) ([Tick](#) x, [Tick](#) y)

Compares two [Tick](#) objects and returns a value indicating whether one is less than, equal to, or greater than the other.

6.367.1 Detailed Description

Provides a mechanism for comparing two [Tick](#) objects.

6.367.2 Member Function Documentation

6.367.2.1 Compare()

```
int Compare (
    Tick x,
    Tick y )
```

Compares two [Tick](#) objects and returns a value indicating whether one is less than, equal to, or greater than the other.

Parameters

<i>x</i>	The first Tick object to compare.
<i>y</i>	The second Tick object to compare.

Returns

A signed integer that indicates the relative values of x and y.

6.368 TickAccumulator Struct Reference

A tick accumulator.

Public Member Functions

- void [AddTicks](#) (int ticks)
Adds a specified number of ticks to the [TickAccumulator](#).
- void [AddTime](#) (double dt, double step, int? maxTicks=null)
Adds a specified amount of time to the [TickAccumulator](#), incrementing the tick count as necessary.
- float [Alpha](#) (double step)
Calculates the alpha value based on the given step.
- bool [ConsumeTick](#) (out bool last)
Consumes a tick from the [TickAccumulator](#).
- void [Start](#) ()
Starts the [TickAccumulator](#), allowing it to accumulate ticks.
- void [Stop](#) ()
Stops the [TickAccumulator](#) from accumulating ticks.

Static Public Member Functions

- static [TickAccumulator StartNew](#) ()
Starts a new [TickAccumulator](#).

Public Attributes

- bool [_running](#)
- double [_scale](#)
- int [_ticks](#)
- double [_time](#)

Properties

- int [Pending](#) [get]
Gets the number of pending ticks in the [TickAccumulator](#).
- double [Remainder](#) [get]
Gets the remaining time in the [TickAccumulator](#).
- bool [Running](#) [get]
Gets a value indicating whether the [TickAccumulator](#) is running.
- double [TimeScale](#) [get, set]
Gets or sets the time scale of the [TickAccumulator](#).

6.368.1 Detailed Description

A tick accumulator.

6.368.2 Member Function Documentation

6.368.2.1 AddTicks()

```
void AddTicks (
    int ticks )
```

Adds a specified number of ticks to the [TickAccumulator](#).

Parameters

<i>ticks</i>	The number of ticks to add.
--------------	-----------------------------

6.368.2.2 AddTime()

```
void AddTime (
    double dt,
    double step,
    int? maxTicks = null )
```

Adds a specified amount of time to the [TickAccumulator](#), incrementing the tick count as necessary.

Parameters

<i>dt</i>	The amount of time to add.
<i>step</i>	The time step value.
<i>maxTicks</i>	The maximum number of ticks to add. If this value is reached, the remaining time is set to 0.

6.368.2.3 Alpha()

```
float Alpha (
    double step )
```

Calculates the alpha value based on the given step.

Parameters

<i>step</i>	The step value used to calculate the alpha.
-------------	---

Returns

The calculated alpha value.

6.368.2.4 ConsumeTick()

```
bool ConsumeTick (
    out bool last )
```

Consumes a tick from the [TickAccumulator](#).

Parameters

<i>last</i>	When this method returns, contains a boolean indicating whether the consumed tick was the last one.
-------------	---

Returns

true if a tick was successfully consumed; otherwise, false.

6.368.2.5 Start()

```
void Start ( )
```

Starts the [TickAccumulator](#), allowing it to accumulate ticks.

6.368.2.6 StartNew()

```
static TickAccumulator StartNew ( ) [static]
```

Starts a new [TickAccumulator](#).

Returns

A new [TickAccumulator](#).

6.368.2.7 Stop()

```
void Stop ( )
```

Stops the [TickAccumulator](#) from accumulating ticks.

6.368.3 Property Documentation

6.368.3.1 Pending

```
int Pending [get]
```

Gets the number of pending ticks in the [TickAccumulator](#).

6.368.3.2 Remainder

```
double Remainder [get]
```

Gets the remaining time in the [TickAccumulator](#).

6.368.3.3 Running

```
bool Running [get]
```

Gets a value indicating whether the [TickAccumulator](#) is running.

6.368.3.4 TimeScale

```
double TimeScale [get], [set]
```

Gets or sets the time scale of the [TickAccumulator](#).

Exceptions

<i>InvalidOperationException</i>	Thrown when the value is less than or equal to 0.
----------------------------------	---

6.369 TickRate Struct Reference

A tick rate is a collection of tick rates.

Classes

- struct [Resolved](#)
Represents a resolved tick rate.
- struct [Selection](#)
Represents a selection of tick rates for client and server.

Public Types

- enum class [ValidateResult](#)
Enumerates the possible results of validating a tick rate selection.

Public Member Functions

- [Selection ClampSelection](#) ([Selection](#) selection)
Clamps the indices in the specified [Selection](#) to valid ranges.
- int [GetDivisor](#) (int index)
Gets the divisor for the tick rate at the specified index.
- int [GetTickRate](#) (int index)
Gets the tick rate at the specified index.
- **TickRate** (params int[] rates)
- int[] [ToArray](#) ()
Converts the tick rates to an array.
- bool [Validate](#) ()
Validates the tick rates in the [TickRate](#).
- [ValidateResult ValidateSelection](#) ([Selection](#) selected)
Validates the tick rates in the specified [Selection](#).

Static Public Member Functions

- static [TickRate Get](#) (int rate)
Retrieves the [TickRate](#) associated with the specified tick rate.
- static void [Init](#) ()
Initializes the [TickRate](#) class by setting up the valid tick rates and their lookup dictionary.
- static void [InitChecked](#) ()
- static bool [IsValid](#) (int rate)
Checks if the provided tick rate is valid.
- static bool [IsValid](#) ([TickRate](#) rate)
Checks if the provided [TickRate](#) is valid.
- static [Resolved Resolve](#) ([Selection](#) selection)
Resolves the specified [Selection](#) into a [Resolved](#) structure.

Public Attributes

- `int _count`
- fixed `int _rates [4]`

Static Public Attributes

- static `Dictionary< int, TickRate > _lookup`
- static `TickRate[] _valid`
- static `ReadOnlyCollection< TickRate > _validReadOnly`

Properties

- static `IReadOnlyList< TickRate > Available` [get]

Gets a read-only list of all available TickRates.
- `int Client` [get]

Gets the tick rate for the client.
- `int Count` [get]

Gets the count of tick rates in the TickRate.
- `int this[int index]` [get]

Gets the tick rate at the specified index.

6.369.1 Detailed Description

A tick rate is a collection of tick rates.

6.369.2 Member Enumeration Documentation

6.369.2.1 ValidateResult

```
enum ValidateResult [strong]
```

Enumerates the possible results of validating a tick rate selection.

Enumerator

<code>Ok</code>	The tick rate selection is valid.
<code>Error</code>	An error occurred during validation.
<code>NotFound</code>	The tick rate selection was not found.
<code>InvalidTickRate</code>	The tick rate selection is invalid.
<code>ServerIndexOutOfRange</code>	The server index in the tick rate selection is out of range.
<code>ClientSendIndexOutOfRange</code>	The client send index in the tick rate selection is out of range.
<code>ServerSendIndexOutOfRange</code>	The server send index in the tick rate selection is out of range.
<code>ServerSendRateLargerThanTickRate</code>	The server send rate in the tick rate selection is larger than the tick rate.

6.369.3 Member Function Documentation

6.369.3.1 ClampSelection()

```
Selection ClampSelection (
    Selection selection )
```

Clamps the indices in the specified [Selection](#) to valid ranges.

Parameters

<i>selection</i>	The Selection to clamp.
------------------	---

Returns

A new [Selection](#) with clamped indices. If the [TickRate](#) is invalid, returns a default [Selection](#).

6.369.3.2 Get()

```
static TickRate Get (
    int rate ) [static]
```

Retrieves the [TickRate](#) associated with the specified tick rate.

Parameters

<i>rate</i>	The tick rate to retrieve the TickRate for.
-------------	---

Returns

The [TickRate](#) associated with the specified tick rate.

Exceptions

<i>InvalidOperationException</i>	Thrown when the tick rate is invalid.
----------------------------------	---------------------------------------

6.369.3.3 GetDivisor()

```
int GetDivisor (
    int index )
```

Gets the divisor for the tick rate at the specified index.

Parameters

<i>index</i>	The index of the tick rate to get the divisor for.
--------------	--

Returns

The divisor for the tick rate at the specified index.

Exceptions

<i>ArgumentOutOfRangeException</i>	Thrown when the index is out of range.
<i>InvalidOperationException</i>	Thrown when the client tick rate is not divisible by the tick rate at the specified index.

6.369.3.4 GetTickRate()

```
int GetTickRate (
    int index )
```

Gets the tick rate at the specified index.

Parameters

<i>index</i>	The index of the tick rate to get.
--------------	------------------------------------

Returns

The tick rate at the specified index.

Exceptions

<i>ArgumentOutOfRangeException</i>	Thrown when the index is out of range.
------------------------------------	--

6.369.3.5 Init()

```
static void Init ( ) [static]
```

Initializes the [TickRate](#) class by setting up the valid tick rates and their lookup dictionary.

6.369.3.6 IsValid() [1/2]

```
static bool IsValid (
    int rate ) [static]
```

Checks if the provided tick rate is valid.

Parameters

<i>rate</i>	The tick rate to validate.
-------------	----------------------------

Returns

true if the tick rate is valid; otherwise, false.

6.369.3.7 IsValid() [2/2]

```
static bool IsValid (
    TickRate rate ) [static]
```

Checks if the provided [TickRate](#) is valid.

Parameters

<i>rate</i>	The TickRate to validate.
-------------	---

Returns

true if the [TickRate](#) is valid; otherwise, false.

6.369.3.8 Resolve()

```
static Resolved Resolve (
    Selection selection ) [static]
```

Resolves the specified [Selection](#) into a [Resolved](#) structure.

Parameters

<i>selection</i>	The Selection to resolve.
------------------	---

Returns

A [Resolved](#) structure that represents the resolved tick rates.

6.369.3.9 ToArray()

```
int [] ToArray ( )
```

Converts the tick rates to an array.

Returns

An array containing the tick rates.

6.369.3.10 Validate()

```
bool Validate ( )
```

Validates the tick rates in the [TickRate](#).

Returns

true if the tick rates are valid; otherwise, false.

The tick rates are valid if:

- There is at least one tick rate.
- There are no more than four tick rates.
- The first tick rate is greater than 0.
- Each tick rate is a divisor of the first tick rate.

6.369.3.11 ValidateSelection()

```
ValidateResult ValidateSelection (
    Selection selected )
```

Validates the tick rates in the specified [Selection](#).

Parameters

<i>selected</i>	The Selection to validate.
-----------------	--

Returns

A `ValidateResult` that indicates the result of the validation.

The `Selection` is valid if:

- The `TickRate` is valid.
- The client tick rate in the `Selection` matches the client tick rate in the `TickRate`.
- The server index in the `Selection` is within the range of available tick rates.
- The server send index in the `Selection` is within the range of available tick rates.
- The client send index in the `Selection` is within the range of available tick rates.
- The server send rate in the `Selection` is not larger than the server tick rate.

6.369.4 Property Documentation

6.369.4.1 Available

```
ICollection<TickRate> Available [static], [get]
```

Gets a read-only list of all available `TickRates`.

This property ensures that the `TickRate` class is initialized before returning the list.

6.369.4.2 Client

```
int Client [get]
```

Gets the tick rate for the client.

6.369.4.3 Count

```
int Count [get]
```

Gets the count of tick rates in the `TickRate`.

6.369.4.4 this[int index]

```
int this[int index] [get]
```

Gets the tick rate at the specified index.

Parameters

<i>index</i>	The index of the tick rate to get.
--------------	------------------------------------

Returns

The tick rate at the specified index.

6.370 TickRate.Resolved Struct Reference

Represents a resolved tick rate.

Public Member Functions

- override string [ToString](#) ()

Public Attributes

- int [Client](#)
The tick rate for the client.
- int [ClientSend](#)
The send tick rate for the client.
- int [Server](#)
The tick rate for the server.
- int [ServerSend](#)
The send tick rate for the server.

Static Public Attributes

- const int [SIZE](#) = 16
The size of the [Resolved](#) structure in bytes.
- const int [WORDS](#) = [SIZE](#) / [Allocator.REPLICATE_WORD_SIZE](#)
The size of the [Resolved](#) structure in words.

Properties

- double [ClientSendDelta](#) [get]
Gets the delta time for the client send rate.
- double [ClientTickDelta](#) [get]
Gets the delta time for the client tick rate.
- int [ClientTickStride](#) [get]
Gets the stride of the client tick rate. This is always 1.
- double [ServerSendDelta](#) [get]
Gets the delta time for the server send rate.
- double [ServerTickDelta](#) [get]
Gets the delta time for the server tick rate.
- int [ServerTickStride](#) [get]
Gets the stride of the server tick rate relative to the client tick rate.

6.370.1 Detailed Description

Represents a resolved tick rate.

The tick rate is resolved by the client and server tick rates.

6.370.2 Member Data Documentation

6.370.2.1 Client

```
int Client
```

The tick rate for the client.

6.370.2.2 ClientSend

```
int ClientSend
```

The send tick rate for the client.

6.370.2.3 Server

```
int Server
```

The tick rate for the server.

6.370.2.4 ServerSend

```
int ServerSend
```

The send tick rate for the server.

6.370.2.5 SIZE

```
const int SIZE = 16 [static]
```

The size of the [Resolved](#) structure in bytes.

6.370.2.6 WORDS

```
const int WORDS = SIZE / Allocator.REPLICATE_WORD_SIZE [static]
```

The size of the [Resolved](#) structure in words.

6.370.3 Property Documentation

6.370.3.1 ClientSendDelta

```
double ClientSendDelta [get]
```

Gets the delta time for the client send rate.

6.370.3.2 ClientTickDelta

```
double ClientTickDelta [get]
```

Gets the delta time for the client tick rate.

6.370.3.3 ClientTickStride

```
int ClientTickStride [get]
```

Gets the stride of the client tick rate. This is always 1.

6.370.3.4 ServerSendDelta

```
double ServerSendDelta [get]
```

Gets the delta time for the server send rate.

6.370.3.5 ServerTickDelta

```
double ServerTickDelta [get]
```

Gets the delta time for the server tick rate.

6.370.3.6 ServerTickStride

```
int ServerTickStride [get]
```

Gets the stride of the server tick rate relative to the client tick rate.

6.371 TickRate.Selection Struct Reference

Represents a selection of tick rates for client and server.

Public Attributes

- int [Client](#)
The tick rate for the client.
- int [ClientSendIndex](#)
The index of the client's send tick rate in the available tick rates.
- int [ServerIndex](#)
The index of the server's tick rate in the available tick rates.
- int [ServerSendIndex](#)
The index of the server's send tick rate in the available tick rates.

6.371.1 Detailed Description

Represents a selection of tick rates for client and server.

6.371.2 Member Data Documentation

6.371.2.1 Client

```
int Client
```

The tick rate for the client.

6.371.2.2 ClientSendIndex

```
int ClientSendIndex
```

The index of the client's send tick rate in the available tick rates.

6.371.2.3 ServerIndex

```
int ServerIndex
```

The index of the server's tick rate in the available tick rates.

6.371.2.4 ServerSendIndex

```
int ServerSendIndex
```

The index of the server's send tick rate in the available tick rates.

6.372 TickTimer Struct Reference

A timer that is based on ticks instead of seconds.

Inherits [INetworkStruct](#).

Public Member Functions

- bool [Expired](#) ([NetworkRunner](#) runner)
Checks if the [TickTimer](#) has expired.
- bool [ExpiredOrNotRunning](#) ([NetworkRunner](#) runner)
Checks if the [TickTimer](#) has expired or is not running.
- int? [RemainingTicks](#) ([NetworkRunner](#) runner)
Gets the number of remaining ticks until the [TickTimer](#) expires.
- float? [RemainingTime](#) ([NetworkRunner](#) runner)
Gets the remaining time in seconds until the [TickTimer](#) expires.
- override string [ToString](#) ()
Returns a string that represents the current [TickTimer](#).

Static Public Member Functions

- static [TickTimer CreateFromSeconds](#) ([NetworkRunner](#) runner, float delayInSeconds)
Creates a [TickTimer](#) from a specified delay in seconds.
- static [TickTimer CreateFromTicks](#) ([NetworkRunner](#) runner, int ticks)
Creates a [TickTimer](#) from a specified number of ticks.

Public Attributes

- int [_target](#)

Properties

- bool [IsRunning](#) [get]
Gets a value indicating whether the [TickTimer](#) is running.
- static [TickTimer None](#) [get]
Gets a [TickTimer](#) that is not running.
- int? [TargetTick](#) [get]
Gets the target tick of the [TickTimer](#).

6.372.1 Detailed Description

A timer that is based on ticks instead of seconds.

6.372.2 Member Function Documentation

6.372.2.1 CreateFromSeconds()

```
static TickTimer CreateFromSeconds (  
    NetworkRunner runner,  
    float delayInSeconds ) [static]
```

Creates a [TickTimer](#) from a specified delay in seconds.

Parameters

<i>runner</i>	The NetworkRunner associated with the TickTimer .
<i>delayInSeconds</i>	The delay in seconds to set the TickTimer to.

Returns

A [TickTimer](#) that will expire after the specified delay in seconds. If the [NetworkRunner](#) is not alive or not running, returns a default [TickTimer](#).

6.372.2.2 CreateFromTicks()

```
static TickTimer CreateFromTicks (  
    NetworkRunner runner,  
    int ticks ) [static]
```

Creates a [TickTimer](#) from a specified number of ticks.

Parameters

<i>runner</i>	The NetworkRunner associated with the TickTimer .
<i>ticks</i>	The number of ticks to set the TickTimer to.

Returns

A [TickTimer](#) that will expire after the specified number of ticks. If the [NetworkRunner](#) is not alive or not running, returns a default [TickTimer](#).

6.372.2.3 Expired()

```
bool Expired (  
    NetworkRunner runner )
```

Checks if the [TickTimer](#) has expired.

Parameters

<i>runner</i>	The NetworkRunner associated with the TickTimer .
---------------	---

Returns

true if the [TickTimer](#) is alive, the runner is running, and the target tick has been reached or passed; otherwise, false.

6.372.2.4 ExpiredOrNotRunning()

```
bool ExpiredOrNotRunning (  
    NetworkRunner runner )
```

Checks if the [TickTimer](#) has expired or is not running.

Parameters

<i>runner</i>	The NetworkRunner associated with the TickTimer .
---------------	---

Returns

true if the [TickTimer](#) is not running, the runner is not running, or the [TickTimer](#) has expired; otherwise, false.

6.372.2.5 RemainingTicks()

```
int? RemainingTicks (
    NetworkRunner runner )
```

Gets the number of remaining ticks until the [TickTimer](#) expires.

Parameters

<i>runner</i>	The NetworkRunner associated with the TickTimer .
---------------	---

Returns

The number of remaining ticks if the [TickTimer](#) is alive and running; otherwise, null.

6.372.2.6 RemainingTime()

```
float? RemainingTime (
    NetworkRunner runner )
```

Gets the remaining time in seconds until the [TickTimer](#) expires.

Parameters

<i>runner</i>	The NetworkRunner associated with the TickTimer .
---------------	---

Returns

The remaining time in seconds if there are remaining ticks; otherwise, null.

6.372.2.7 ToString()

```
override string ToString ( )
```

Returns a string that represents the current [TickTimer](#).

Returns

A string that represents the current [TickTimer](#).

6.372.3 Property Documentation

6.372.3.1 IsRunning

```
bool IsRunning [get]
```

Gets a value indicating whether the [TickTimer](#) is running.

true if the [TickTimer](#) is running; otherwise, false.

6.372.3.2 None

```
TickTimer None [static], [get]
```

Gets a [TickTimer](#) that is not running.

6.372.3.3 TargetTick

```
int? TargetTick [get]
```

Gets the target tick of the [TickTimer](#).

The target tick if the [TickTimer](#) is running; otherwise, null.

6.373 Timer Struct Reference

Represents a high-resolution timer.

Public Member Functions

- long **GetDelta** ()
- void **Reset** ()
Resets the timer to its initial state.
- void **Restart** ()
Restarts the timer, setting the elapsed time to zero and starting it.
- void **Start** ()
Starts the timer if it is not already running.
- void **Stop** ()
Stops the timer if it is running and updates the elapsed time.

Static Public Member Functions

- static [Timer](#) **StartNew** ()
Creates and starts a new timer.

Public Attributes

- `long _elapsed`
- `byte _running`
- `long _start`

Properties

- double `ElapsedInMilliseconds` [get]
Gets the elapsed time in milliseconds.
- double `ElapsedInSeconds` [get]
Gets the elapsed time in seconds.
- long? `ElapsedInTicks` [get]
Gets the elapsed time in ticks.
- bool `IsRunning` [get]
Gets a value indicating whether the timer is running.

6.373.1 Detailed Description

Represents a high-resolution timer.

6.373.2 Member Function Documentation

6.373.2.1 Reset()

```
void Reset ( )
```

Resets the timer to its initial state.

6.373.2.2 Restart()

```
void Restart ( )
```

Restarts the timer, setting the elapsed time to zero and starting it.

6.373.2.3 Start()

```
void Start ( )
```

Starts the timer if it is not already running.

6.373.2.4 StartNew()

```
static Timer StartNew ( ) [static]
```

Creates and starts a new timer.

Returns

A new instance of the [Timer](#) struct.

6.373.2.5 Stop()

```
void Stop ( )
```

Stops the timer if it is running and updates the elapsed time.

6.373.3 Property Documentation

6.373.3.1 ElapsedInMilliseconds

```
double ElapsedInMilliseconds [get]
```

Gets the elapsed time in milliseconds.

6.373.3.2 ElapsedInSeconds

```
double ElapsedInSeconds [get]
```

Gets the elapsed time in seconds.

6.373.3.3 ElapsedInTicks

```
long? ElapsedInTicks [get]
```

Gets the elapsed time in ticks.

6.373.3.4 IsRunning

```
bool IsRunning [get]
```

Gets a value indicating whether the timer is running.

6.374 TimeSyncConfiguration Class Reference

Time Synchronization Configuration

Public Attributes

- double [MaxLateInputs](#) = 1.0
- double [MaxLateSnapshots](#) = 1.0
- int [RedundantInputs](#) = 1
- int [RedundantSnapshots](#) = 1
- double [SampleWindowSeconds](#) = 1.0

6.374.1 Detailed Description

Time Synchronization Configuration

6.374.2 Member Data Documentation

6.374.2.1 MaxLateInputs

```
double MaxLateInputs = 1.0
```

The maximum percentage of inputs that should ever arrive late because of jitter.

Decreasing this increases the client's simulation offset.

6.374.2.2 MaxLateSnapshots

```
double MaxLateSnapshots = 1.0
```

The maximum percentage of snapshots that should ever arrive late because of jitter.

Decreasing this increases the client's interpolation delay.

6.374.2.3 RedundantInputs

```
int RedundantInputs = 1
```

The number of consecutive, additional chances each input should have to reach the server before it's needed.

Increasing this increases the client's simulation offset.

6.374.2.4 RedundantSnapshots

```
int RedundantSnapshots = 1
```

The number of consecutive snapshots a client should be able to miss without disrupting their interpolation.

Increasing this increases the client's interpolation delay.

6.374.2.5 SampleWindowSeconds

```
double SampleWindowSeconds = 1.0
```

How big of a window the client looks at to evaluate the latest network conditions.

Increasing this makes the client slower to adapt to changes.

6.375 ToggleLeftAttribute Class Reference

Specifies that the bool field should be drawn as the toggle on the left side of the label.

Inherits [DrawerPropertyAttribute](#).

6.375.1 Detailed Description

Specifies that the bool field should be drawn as the toggle on the left side of the label.

6.376 UnitAttribute Class Reference

Unit Attribute class. Used to mark a field with the respective [Units](#)

Inherits [DecoratingPropertyAttribute](#).

Public Member Functions

- [UnitAttribute](#) ([Units](#) units)

Initializes a new instance of the [UnitAttribute](#) class with the specified unit.

Properties

- [Units Unit](#) [get]

Selected Unit for the field.

Additional Inherited Members

6.376.1 Detailed Description

Unit Attribute class. Used to mark a field with the respective [Units](#)

6.376.2 Constructor & Destructor Documentation

6.376.2.1 UnitAttribute()

```
UnitAttribute (  
    Units units )
```

Initializes a new instance of the [UnitAttribute](#) class with the specified unit.

Parameters

<i>units</i>	
--------------	--

6.376.3 Property Documentation

6.376.3.1 Unit

```
Units Unit [get]
```

Selected Unit for the field.

6.377 UnityAddressablesRuntimeKeyAttribute Class Reference

Specifies that the string field represents a key for Unity Addressables.

Inherits [PropertyAttribute](#).

6.377.1 Detailed Description

Specifies that the string field represents a key for Unity Addressables.

6.378 UnityAssetGuidAttribute Class Reference

Specifies that the string field represents a GUID of an asset.

Inherits [DrawerPropertyAttribute](#).

6.378.1 Detailed Description

Specifies that the string field represents a GUID of an asset.

6.379 UnityContextMenuAttribute Class Reference

Unity ContextMenuAttribute

Inherits [Attribute](#).

Public Member Functions

- [UnityContextMenuAttribute](#) (string function, string name)
ContextMenuAttribute Constructor

Properties

- int [order](#) [get, set]
ContextMenuAttribute Order

6.379.1 Detailed Description

Unity ContextMenuAttribute

6.379.2 Constructor & Destructor Documentation

6.379.2.1 UnityContextMenuAttribute()

```
UnityContextMenuAttribute (
    string function,
    string name )
```

ContextMenuAttribute Constructor

6.379.3 Property Documentation

6.379.3.1 order

```
int order [get], [set]
```

ContextMenuAttribute Order

6.380 UnityDelayedAttribute Class Reference

Unity DelayedAttribute

Inherits Attribute.

Properties

- int `order` [get, set]
DelayedAttribute Order

6.380.1 Detailed Description

Unity DelayedAttribute

6.380.2 Property Documentation

6.380.2.1 order

```
int order [get], [set]
```

DelayedAttribute Order

6.381 UnityFormerlySerializedAsAttribute Class Reference

Unity FormerlySerializedAsAttribute

Inherits Attribute.

Public Member Functions

- [UnityFormerlySerializedAsAttribute](#) (string oldName)
FormerlySerializedAsAttribute Constructor

6.381.1 Detailed Description

Unity FormerlySerializedAsAttribute

6.381.2 Constructor & Destructor Documentation

6.381.2.1 UnityFormerlySerializedAsAttribute()

```
UnityFormerlySerializedAsAttribute (  
    string oldName )
```

FormerlySerializedAsAttribute Constructor

6.382 UnityHeaderAttribute Class Reference

Unity HeaderAttribute

Inherits Attribute.

Public Member Functions

- [UnityHeaderAttribute](#) (string header)
HeaderAttribute Constructor

Properties

- int [order](#) [get, set]
HeaderAttribute Order

6.382.1 Detailed Description

Unity HeaderAttribute

6.382.2 Constructor & Destructor Documentation

6.382.2.1 UnityHeaderAttribute()

```
UnityHeaderAttribute (  
    string header )
```

HeaderAttribute Constructor

6.382.3 Property Documentation

6.382.3.1 order

```
int order [get], [set]
```

HeaderAttribute Order

6.383 UnityMinAttribute Class Reference

Unity MinAttribute

Inherits Attribute.

Public Member Functions

- [UnityMinAttribute](#) (float min)
MinAttribute Constructor

Properties

- int [order](#) [get, set]
MinAttribute Order

6.383.1 Detailed Description

Unity MinAttribute

6.383.2 Constructor & Destructor Documentation

6.383.2.1 UnityMinAttribute()

```
UnityMinAttribute (  
    float min )
```

MinAttribute Constructor

6.383.3 Property Documentation

6.383.3.1 order

```
int order [get], [set]
```

MinAttribute Order

6.384 UnityMultilineAttribute Class Reference

Unity MultilineAttribute

Inherits Attribute.

Properties

- int `order` [get, set]
MultilineAttribute Order

6.384.1 Detailed Description

Unity MultilineAttribute

6.384.2 Property Documentation

6.384.2.1 order

```
int order [get], [set]
```

MultilineAttribute Order

6.385 UnityNonReorderableAttribute Class Reference

Unity NonReorderableAttribute

Inherits Attribute.

Properties

- int [order](#) [get, set]
NonReorderableAttribute Order

6.385.1 Detailed Description

Unity NonReorderableAttribute

6.385.2 Property Documentation

6.385.2.1 order

```
int order [get], [set]
```

NonReorderableAttribute Order

6.386 UnityNonSerializedAttribute Class Reference

Unity NonSerializedAttribute

Inherits Attribute.

6.386.1 Detailed Description

Unity NonSerializedAttribute

6.387 UnityRangeAttribute Class Reference

Unity RangeAttribute

Inherits Attribute.

Public Member Functions

- [UnityRangeAttribute](#) (float min, float max)
RangeAttribute Constructor

Properties

- int [order](#) [get, set]
RangeAttribute Order

6.387.1 Detailed Description

Unity RangeAttribute

6.387.2 Constructor & Destructor Documentation

6.387.2.1 UnityRangeAttribute()

```
UnityRangeAttribute (  
    float min,  
    float max )
```

RangeAttribute Constructor

6.387.3 Property Documentation

6.387.3.1 order

```
int order [get], [set]
```

RangeAttribute Order

6.388 UnityResourcePathAttribute Class Reference

Specifies that the string field represents a path to a Unity resource.

Inherits [DrawerPropertyAttribute](#).

Public Member Functions

- [UnityResourcePathAttribute](#) (Type resourceType)
Initializes a new instance of the class with the specified resource type.

Properties

- Type [ResourceType](#) [get]
The type of the resource.

6.388.1 Detailed Description

Specifies that the string field represents a path to a Unity resource.

6.388.2 Constructor & Destructor Documentation

6.388.2.1 UnityResourcePathAttribute()

```
UnityResourcePathAttribute (  
    Type resourceType )
```

Initializes a new instance of the class with the specified resource type.

Parameters

<i>resourceType</i>

6.388.3 Property Documentation

6.388.3.1 ResourceType

```
Type ResourceType [get]
```

The type of the resource.

6.389 UnitySerializeField Class Reference

Unity SerializeField

Inherits Attribute.

6.389.1 Detailed Description

Unity SerializeField

6.390 UnitySerializeReference Class Reference

Unity SerializeReference

Inherits Attribute.

6.390.1 Detailed Description

Unity SerializeReference

6.391 UnitySpaceAttribute Class Reference

Unity SpaceAttribute

Inherits Attribute.

Public Member Functions

- [UnitySpaceAttribute](#) (float space)
SpaceAttribute Constructor

Properties

- int [order](#) [get, set]
SpaceAttribute Order

6.391.1 Detailed Description

Unity SpaceAttribute

6.391.2 Constructor & Destructor Documentation

6.391.2.1 UnitySpaceAttribute()

```
UnitySpaceAttribute (  
    float space )
```

SpaceAttribute Constructor

6.391.3 Property Documentation

6.391.3.1 order

```
int order [get], [set]
```

SpaceAttribute Order

6.392 UnityTooltipAttribute Class Reference

Unity TooltipAttribute

Inherits Attribute.

Public Member Functions

- [UnityTooltipAttribute](#) (string tooltip)
TooltipAttribute Constructor

Properties

- int [order](#) [get, set]
TooltipAttribute Order

6.392.1 Detailed Description

Unity TooltipAttribute

6.392.2 Constructor & Destructor Documentation

6.392.2.1 UnityTooltipAttribute()

```
UnityTooltipAttribute (  
    string tooltip )
```

TooltipAttribute Constructor

6.392.3 Property Documentation

6.392.3.1 order

```
int order [get], [set]
```

TooltipAttribute Order

6.393 UTF32Tools Class Reference

[UTF32Tools](#) provides a set of methods to work with UTF32 encoded strings.

Classes

- struct [CharEnumerator](#)
Enumerates the characters in a UTF-32 encoded string.
- struct [ConversionResult](#)
Represents the result of a conversion operation, containing the number of characters and code points processed.

Static Public Member Functions

- static [ConversionResult Convert](#) (char *str, int strLength, uint *dst, int dstCapacity)
Converts a UTF-16 encoded string to a UTF-32 encoded representation.
- static [ConversionResult Convert](#) (string str, uint *dst, int dstCapacity)
Converts a UTF-16 encoded string to a UTF-32 encoded representation.
- static int [GetLength](#) (string str)
Gets the length of a UTF-32 encoded string.

6.393.1 Detailed Description

[UTF32Tools](#) provides a set of methods to work with UTF32 encoded strings.

6.393.2 Member Function Documentation

6.393.2.1 Convert() [1/2]

```
static ConversionResult Convert (  
    char * str,  
    int strLength,  
    uint * dst,  
    int dstCapacity ) [static]
```

Converts a UTF-16 encoded string to a UTF-32 encoded representation.

Parameters

<i>str</i>	A pointer to the UTF-16 encoded string to convert.
<i>strLength</i>	The length of the UTF-16 encoded string.
<i>dst</i>	A pointer to the destination buffer where the UTF-32 encoded result will be stored.
<i>dstCapacity</i>	The capacity of the destination buffer.

Returns

A [ConversionResult](#) containing the number of characters and code points processed.

6.393.2.2 Convert() [2/2]

```
static ConversionResult Convert (  
    string str,  
    uint * dst,  
    int dstCapacity ) [static]
```

Converts a UTF-16 encoded string to a UTF-32 encoded representation.

Parameters

<i>str</i>	The UTF-16 encoded string to convert.
<i>dst</i>	A pointer to the destination buffer where the UTF-32 encoded result will be stored.
<i>dstCapacity</i>	The capacity of the destination buffer.

Returns

A [ConversionResult](#) containing the number of characters and code points processed.

6.393.2.3 GetLength()

```
static int GetLength (  
    string str ) [static]
```

Gets the length of a UTF-32 encoded string.

Parameters

<i>str</i>	The UTF-16 encoded string to measure.
------------	---------------------------------------

Returns

The length of the UTF-32 encoded string.

6.394 UTF32Tools.CharEnumerator Struct Reference

Enumerates the characters in a UTF-32 encoded string.

Inherits `IEnumerator< char >`.

Public Member Functions

- void [Dispose](#) ()
Releases all resources used by the [CharEnumerator](#).
- bool [MoveNext](#) ()
Advances the enumerator to the next character in the UTF-32 encoded string.
- void [Reset](#) ()
Sets the enumerator to its initial position, which is before the first character in the UTF-32 encoded string.

Properties

- char [Current](#) [get]
Gets the current character in the enumeration.
- object `System.Collections.IEnumerator`. [Current](#) [get]

6.394.1 Detailed Description

Enumerates the characters in a UTF-32 encoded string.

6.394.2 Member Function Documentation

6.394.2.1 Dispose()

```
void Dispose ( )
```

Releases all resources used by the [CharEnumerator](#).

6.394.2.2 MoveNext()

```
bool MoveNext ( )
```

Advances the enumerator to the next character in the UTF-32 encoded string.

Returns

`true` if the enumerator was successfully advanced to the next character; `false` if the enumerator has passed the end of the string.

6.394.2.3 Reset()

```
void Reset ( )
```

Sets the enumerator to its initial position, which is before the first character in the UTF-32 encoded string.

6.394.3 Property Documentation

6.394.3.1 Current

```
char Current [get]
```

Gets the current character in the enumeration.

6.395 UTF32Tools.ConversionResult Struct Reference

Represents the result of a conversion operation, containing the number of characters and code points processed.

Public Member Functions

- [ConversionResult](#) (int words, int characters)
Initializes a new instance of the [ConversionResult](#) struct.

Public Attributes

- readonly int [CharacterCount](#)
Gets the number of characters processed.
- readonly int [CodePointCount](#)
Gets the number of code points processed.

6.395.1 Detailed Description

Represents the result of a conversion operation, containing the number of characters and code points processed.

6.395.2 Constructor & Destructor Documentation

6.395.2.1 ConversionResult()

```
ConversionResult (
    int words,
    int characters )
```

Initializes a new instance of the [ConversionResult](#) struct.

Parameters

<i>words</i>	The number of code points processed.
<i>characters</i>	The number of characters processed.

6.395.3 Member Data Documentation

6.395.3.1 CharacterCount

```
readonly int CharacterCount
```

Gets the number of characters processed.

6.395.3.2 CodePointCount

```
readonly int CodePointCount
```

Gets the number of code points processed.

6.396 Vector2Compressed Struct Reference

Represents a compressed Vector2 value for network transmission.

Inherits [INetworkStruct](#), and [IEquatable< Vector2Compressed >](#).

Public Member Functions

- override bool [Equals](#) (object obj)
Checks if the provided object is a [Vector2Compressed](#) instance and if it's equal to the current [Vector2Compressed](#) instance.
- bool [Equals](#) ([Vector2Compressed](#) other)
Checks if the current [Vector2Compressed](#) instance is equal to the other [Vector2Compressed](#) instance.
- override int [GetHashCode](#) ()
Returns the hash code for the current [Vector2Compressed](#) instance.

Static Public Member Functions

- static implicit operator [Vector2](#) ([Vector2Compressed](#) q)
Implicit conversion from [Vector2Compressed](#) to [Vector2](#).
- static implicit operator [Vector2Compressed](#) ([Vector2](#) v)
Implicit conversion from [Vector2](#) to [Vector2Compressed](#).
- static bool operator [!=](#) ([Vector2Compressed](#) left, [Vector2Compressed](#) right)
Inequality operator for [Vector2Compressed](#) struct.
- static bool operator [==](#) ([Vector2Compressed](#) left, [Vector2Compressed](#) right)
Equality operator for [Vector2Compressed](#) struct.

Public Attributes

- int [xEncoded](#)
Encoded value of the x component.
- int [yEncoded](#)
Encoded value of the y component.

Properties

- float [X](#) [get, set]
Gets or sets the x component.
- float [Y](#) [get, set]
Gets or sets the y component.

6.396.1 Detailed Description

Represents a compressed [Vector2](#) value for network transmission.

6.396.2 Member Function Documentation

6.396.2.1 Equals() [1/2]

```
override bool Equals (  
    object obj )
```

Checks if the provided object is a [Vector2Compressed](#) instance and if it's equal to the current [Vector2Compressed](#) instance.

Parameters

<i>obj</i>	The object to compare with the current Vector2Compressed instance.
------------	--

Returns

True if the provided object is a [Vector2Compressed](#) instance and it's equal to the current [Vector2Compressed](#) instance, otherwise false.

6.396.2.2 Equals() [2/2]

```
bool Equals (
    Vector2Compressed other )
```

Checks if the current [Vector2Compressed](#) instance is equal to the other [Vector2Compressed](#) instance.

Parameters

<i>other</i>	The other Vector2Compressed instance to compare with the current Vector2Compressed instance.
--------------	--

Returns

True if the values of both [Vector2Compressed](#) instances are equal, otherwise false.

6.396.2.3 GetHashCode()

```
override int GetHashCode ( )
```

Returns the hash code for the current [Vector2Compressed](#) instance.

Returns

A hash code for the current [Vector2Compressed](#) instance.

6.396.2.4 operator Vector2()

```
static implicit operator Vector2 (
    Vector2Compressed q ) [static]
```

Implicit conversion from [Vector2Compressed](#) to [Vector2](#).

Parameters

<i>q</i>	The Vector2Compressed instance to convert.
----------	--

Returns

The decompressed Vector2 value of the [Vector2Compressed](#) instance.

6.396.2.5 operator Vector2Compressed()

```
static implicit operator Vector2Compressed (  
    Vector2 v ) [static]
```

Implicit conversion from Vector2 to [Vector2Compressed](#).

Parameters

<i>v</i>	The Vector2 value to convert.
----------	-------------------------------

Returns

A new [Vector2Compressed](#) instance with the compressed value of the Vector2.

6.396.2.6 operator"!=(())

```
static bool operator!= (  
    Vector2Compressed left,  
    Vector2Compressed right ) [static]
```

Inequality operator for [Vector2Compressed](#) struct.

Parameters

<i>left</i>	First Vector2Compressed instance.
<i>right</i>	Second Vector2Compressed instance.

Returns

True if the value of the first [Vector2Compressed](#) instance is not equal to the value of the second [Vector2Compressed](#) instance, otherwise false.

6.396.2.7 operator==()

```
static bool operator== (
    Vector2Compressed left,
    Vector2Compressed right ) [static]
```

Equality operator for [Vector2Compressed](#) struct.

Parameters

<i>left</i>	First Vector2Compressed instance.
<i>right</i>	Second Vector2Compressed instance.

Returns

True if the value of the first [Vector2Compressed](#) instance is equal to the value of the second [Vector2Compressed](#) instance, otherwise false.

6.396.3 Member Data Documentation

6.396.3.1 xEncoded

```
int xEncoded
```

Encoded value of the x component.

6.396.3.2 yEncoded

```
int yEncoded
```

Encoded value of the y component.

6.396.4 Property Documentation

6.396.4.1 X

```
float X [get], [set]
```

Gets or sets the x component.

6.396.4.2 Y

```
float Y [get], [set]
```

Gets or sets the y component.

6.397 Vector3Compressed Struct Reference

Represents a compressed Vector3 value for network transmission.

Inherits [INetworkStruct](#), and [IEquatable< Vector3Compressed >](#).

Public Member Functions

- override bool [Equals](#) (object obj)
Checks if the provided object is a [Vector3Compressed](#) instance and if it's equal to the current [Vector3Compressed](#) instance.
- bool [Equals](#) ([Vector3Compressed](#) other)
Checks if the current [Vector3Compressed](#) instance is equal to the other [Vector3Compressed](#) instance.
- override int [GetHashCode](#) ()
Returns the hash code for the current [Vector3Compressed](#) instance.

Static Public Member Functions

- static implicit [operator Vector2](#) ([Vector3Compressed](#) q)
Implicit conversion from [Vector3Compressed](#) to [Vector2](#).
- static implicit [operator Vector3](#) ([Vector3Compressed](#) q)
Implicit conversion from [Vector3Compressed](#) to [Vector3](#).
- static implicit [operator Vector3Compressed](#) ([Vector2](#) v)
Implicit conversion from [Vector2](#) to [Vector3Compressed](#).
- static implicit [operator Vector3Compressed](#) ([Vector3](#) v)
Implicit conversion from [Vector3](#) to [Vector3Compressed](#).
- static bool [operator!=](#) ([Vector3Compressed](#) left, [Vector3Compressed](#) right)
Inequality operator for [Vector3Compressed](#) struct.
- static bool [operator==](#) ([Vector3Compressed](#) left, [Vector3Compressed](#) right)
Equality operator for [Vector3Compressed](#) struct.

Public Attributes

- int [xEncoded](#)
Encoded value of the x component.
- int [yEncoded](#)
Encoded value of the y component.
- int [zEncoded](#)
Encoded value of the z component.

Properties

- float **X** [get, set]
Gets or sets the x component.
- float **Y** [get, set]
Gets or sets the y component.
- float **Z** [get, set]
Gets or sets the z component.

6.397.1 Detailed Description

Represents a compressed Vector3 value for network transmission.

6.397.2 Member Function Documentation

6.397.2.1 Equals() [1/2]

```
override bool Equals (  
    object obj )
```

Checks if the provided object is a [Vector3Compressed](#) instance and if it's equal to the current [Vector3Compressed](#) instance.

Parameters

<i>obj</i>	The object to compare with the current Vector3Compressed instance.
------------	--

Returns

True if the provided object is a [Vector3Compressed](#) instance and it's equal to the current [Vector3Compressed](#) instance, otherwise false.

6.397.2.2 Equals() [2/2]

```
bool Equals (  
    Vector3Compressed other )
```

Checks if the current [Vector3Compressed](#) instance is equal to the other [Vector3Compressed](#) instance.

Parameters

<i>other</i>	The other Vector3Compressed instance to compare with the current Vector3Compressed instance.
--------------	--

Returns

True if the values of both [Vector3Compressed](#) instances are equal, otherwise false.

6.397.2.3 GetHashCode()

```
override int GetHashCode ( )
```

Returns the hash code for the current [Vector3Compressed](#) instance.

Returns

A hash code for the current [Vector3Compressed](#) instance.

6.397.2.4 operator Vector2()

```
static implicit operator Vector2 (
    Vector3Compressed q ) [static]
```

Implicit conversion from [Vector3Compressed](#) to Vector2.

Parameters

<i>q</i>	The Vector3Compressed instance to convert.
----------	--

Returns

The decompressed Vector2 value of the [Vector3Compressed](#) instance.

6.397.2.5 operator Vector3()

```
static implicit operator Vector3 (
    Vector3Compressed q ) [static]
```

Implicit conversion from [Vector3Compressed](#) to Vector3.

Parameters

<i>q</i>	The Vector3Compressed instance to convert.
----------	--

Returns

The decompressed Vector3 value of the [Vector3Compressed](#) instance.

6.397.2.6 operator Vector3Compressed() [1/2]

```
static implicit operator Vector3Compressed (  
    Vector2 v ) [static]
```

Implicit conversion from Vector2 to [Vector3Compressed](#).

Parameters

v	The Vector2 value to convert.
---	-------------------------------

Returns

A new [Vector3Compressed](#) instance with the compressed value of the Vector2.

6.397.2.7 operator Vector3Compressed() [2/2]

```
static implicit operator Vector3Compressed (  
    Vector3 v ) [static]
```

Implicit conversion from Vector3 to [Vector3Compressed](#).

Parameters

v	The Vector3 value to convert.
---	-------------------------------

Returns

A new [Vector3Compressed](#) instance with the compressed value of the Vector3.

6.397.2.8 operator"!=(())

```
static bool operator!= (  
    Vector3Compressed left,  
    Vector3Compressed right ) [static]
```

Inequality operator for [Vector3Compressed](#) struct.

Parameters

<i>left</i>	First Vector3Compressed instance.
<i>right</i>	Second Vector3Compressed instance.

Returns

True if the value of the first [Vector3Compressed](#) instance is not equal to the value of the second [Vector3Compressed](#) instance, otherwise false.

6.397.2.9 operator==()

```
static bool operator== (
    Vector3Compressed left,
    Vector3Compressed right ) [static]
```

Equality operator for [Vector3Compressed](#) struct.

Parameters

<i>left</i>	First Vector3Compressed instance.
<i>right</i>	Second Vector3Compressed instance.

Returns

True if the value of the first [Vector3Compressed](#) instance is equal to the value of the second [Vector3Compressed](#) instance, otherwise false.

6.397.3 Member Data Documentation**6.397.3.1 xEncoded**

```
int xEncoded
```

Encoded value of the x component.

6.397.3.2 yEncoded

```
int yEncoded
```

Encoded value of the y component.

6.397.3.3 zEncoded

```
int zEncoded
```

Encoded value of the z component.

6.397.4 Property Documentation

6.397.4.1 X

```
float X [get], [set]
```

Gets or sets the x component.

6.397.4.2 Y

```
float Y [get], [set]
```

Gets or sets the y component.

6.397.4.3 Z

```
float Z [get], [set]
```

Gets or sets the z component.

6.398 Vector4Compressed Struct Reference

Represents a compressed Vector4 value for network transmission.

Inherits [INetworkStruct](#), and [IEquatable< Vector4Compressed >](#).

Public Member Functions

- override bool [Equals](#) (object obj)
Checks if the provided object is a [Vector4Compressed](#) instance and if it's equal to the current [Vector4Compressed](#) instance.
- bool [Equals](#) ([Vector4Compressed](#) other)
Checks if the current [Vector4Compressed](#) instance is equal to the other [Vector4Compressed](#) instance.
- override int [GetHashCode](#) ()
Returns the hash code for the current [Vector4Compressed](#) instance.

Static Public Member Functions

- static implicit [operator Vector4](#) ([Vector4Compressed](#) q)
Implicit conversion from [Vector4Compressed](#) to [Vector4](#).
- static implicit [operator Vector4Compressed](#) ([Vector4](#) v)
Implicit conversion from [Vector4](#) to [Vector4Compressed](#).
- static bool [operator!=](#) ([Vector4Compressed](#) left, [Vector4Compressed](#) right)
Inequality operator for [Vector4Compressed](#) struct.
- static bool [operator==](#) ([Vector4Compressed](#) left, [Vector4Compressed](#) right)
Equality operator for [Vector4Compressed](#) struct.

Public Attributes

- int [wEncoded](#)
Encoded value of the w component.
- int [xEncoded](#)
Encoded value of the x component.
- int [yEncoded](#)
Encoded value of the y component.
- int [zEncoded](#)
Encoded value of the z component.

Properties

- float [W](#) [get, set]
Gets or sets the w component.
- float [X](#) [get, set]
Gets or sets the x component.
- float [Y](#) [get, set]
Gets or sets the y component.
- float [Z](#) [get, set]
Gets or sets the z component.

6.398.1 Detailed Description

Represents a compressed [Vector4](#) value for network transmission.

6.398.2 Member Function Documentation

6.398.2.1 Equals() [1/2]

```
override bool Equals (  
    object obj )
```

Checks if the provided object is a [Vector4Compressed](#) instance and if it's equal to the current [Vector4Compressed](#) instance.

Parameters

<i>obj</i>	The object to compare with the current Vector4Compressed instance.
------------	--

Returns

True if the provided object is a [Vector4Compressed](#) instance and it's equal to the current [Vector4Compressed](#) instance, otherwise false.

6.398.2.2 Equals() [2/2]

```
bool Equals (
    Vector4Compressed other )
```

Checks if the current [Vector4Compressed](#) instance is equal to the other [Vector4Compressed](#) instance.

Parameters

<i>other</i>	The other Vector4Compressed instance to compare with the current Vector4Compressed instance.
--------------	--

Returns

True if the values of both [Vector4Compressed](#) instances are equal, otherwise false.

6.398.2.3 GetHashCode()

```
override int GetHashCode ( )
```

Returns the hash code for the current [Vector4Compressed](#) instance.

Returns

A hash code for the current [Vector4Compressed](#) instance.

6.398.2.4 operator Vector4()

```
static implicit operator Vector4 (
    Vector4Compressed q ) [static]
```

Implicit conversion from [Vector4Compressed](#) to [Vector4](#).

Parameters

<i>q</i>	The Vector4Compressed instance to convert.
----------	--

Returns

The decompressed Vector4 value of the [Vector4Compressed](#) instance.

6.398.2.5 operator Vector4Compressed()

```
static implicit operator Vector4Compressed (  
    Vector4 v ) [static]
```

Implicit conversion from Vector4 to [Vector4Compressed](#).

Parameters

<i>v</i>	The Vector4 value to convert.
----------	-------------------------------

Returns

A new [Vector4Compressed](#) instance with the compressed value of the Vector4.

6.398.2.6 operator"!=(())

```
static bool operator!= (  
    Vector4Compressed left,  
    Vector4Compressed right ) [static]
```

Inequality operator for [Vector4Compressed](#) struct.

Parameters

<i>left</i>	First Vector4Compressed instance.
<i>right</i>	Second Vector4Compressed instance.

Returns

True if the value of the first [Vector4Compressed](#) instance is not equal to the value of the second [Vector4Compressed](#) instance, otherwise false.

6.398.2.7 operator==()

```
static bool operator== (
    Vector4Compressed left,
    Vector4Compressed right ) [static]
```

Equality operator for [Vector4Compressed](#) struct.

Parameters

<i>left</i>	First Vector4Compressed instance.
<i>right</i>	Second Vector4Compressed instance.

Returns

True if the value of the first [Vector4Compressed](#) instance is equal to the value of the second [Vector4Compressed](#) instance, otherwise false.

6.398.3 Member Data Documentation

6.398.3.1 wEncoded

```
int wEncoded
```

Encoded value of the w component.

6.398.3.2 xEncoded

```
int xEncoded
```

Encoded value of the x component.

6.398.3.3 yEncoded

```
int yEncoded
```

Encoded value of the y component.

6.398.3.4 zEncoded

```
int zEncoded
```

Encoded value of the z component.

6.398.4 Property Documentation

6.398.4.1 W

```
float W [get], [set]
```

Gets or sets the w component.

6.398.4.2 X

```
float X [get], [set]
```

Gets or sets the x component.

6.398.4.3 Y

```
float Y [get], [set]
```

Gets or sets the y component.

6.398.4.4 Z

```
float Z [get], [set]
```

Gets or sets the z component.

6.399 Versioning Class Reference

The [Versioning](#) class provides methods and properties related to versioning.

Static Public Member Functions

- static string [AssemblyFileVersion](#) ()
Gets the assembly version as a string.
- static Version [ShortVersion](#) (this Version version)
Get the short version of the version.

Static Public Attributes

- static readonly Version [InvalidVersion](#) = new Version(0, 0, 0)
Represents an invalid version with all components set to zero.

Properties

- static Version [GetCurrentVersion](#) [get]
Gets the current version of the assembly.

6.399.1 Detailed Description

The [Versioning](#) class provides methods and properties related to versioning.

6.399.2 Member Function Documentation

6.399.2.1 AssemblyFileVersion()

```
static string AssemblyFileVersion ( ) [static]
```

Gets the assembly version as a string.

Returns

The assembly version as a string.

6.399.2.2 ShortVersion()

```
static Version ShortVersion (  
    this Version version ) [static]
```

Get the short version of the version.

Parameters

<code>version</code>	The version to get the short version of.
----------------------	--

6.399.3 Member Data Documentation

6.399.3.1 InvalidVersion

```
readonly Version InvalidVersion = new Version(0, 0, 0) [static]
```

Represents an invalid version with all components set to zero.

6.399.4 Property Documentation

6.399.4.1 GetCurrentVersion

```
Version GetCurrentVersion [static], [get]
```

Gets the current version of the assembly.

6.400 WarnIfAttribute Class Reference

Editor attribute for adding notices to fields if the condition member evaluates as `true`. Condition member can be a property, field or method (with a return value).

Inherits [DofAttributeBase](#).

Public Member Functions

- [WarnIfAttribute](#) (string conditionMember, bool compareToValue, string message, [CompareOperator](#) compare=[CompareOperator.Equal](#))
- [WarnIfAttribute](#) (string conditionMember, double compareToValue, string message, [CompareOperator](#) compare=[CompareOperator.Equal](#))
- [WarnIfAttribute](#) (string conditionMember, long compareToValue, string message, [CompareOperator](#) compare=[CompareOperator.Equal](#))
- [WarnIfAttribute](#) (string conditionMember, string message)

Initializes a new instance that will hide the field if the condition member is not equal to zero.

Public Attributes

- bool [AsBox](#)
Should the warning be shown as a box?
- string [Message](#)
The default warning text, when a warning is shown.

Additional Inherited Members

6.400.1 Detailed Description

Editor attribute for adding notices to fields if the condition member evaluates as `true`. Condition member can be a property, field or method (with a return value).

6.400.2 Constructor & Destructor Documentation

6.400.2.1 WarnIfAttribute()

```
WarnIfAttribute (
    string conditionMember,
    string message )
```

Initializes a new instance that will hide the field if the condition member is not equal to zero.

Parameters

<i>conditionMember</i>	
<i>message</i>	

6.400.3 Member Data Documentation

6.400.3.1 AsBox

```
bool AsBox
```

Should the warning be shown as a box?

6.400.3.2 Message

```
string Message
```

The default warning text, when a warning is shown.

6.401 WeaverGeneratedAttribute Class Reference

Weaver Generated Attribute

Inherits Attribute.

6.401.1 Detailed Description

Weaver Generated Attribute

6.402 IExportedWordCount Interface Reference

Used in plugin. Indicates that NetworkBehaviour.DynamicWordCount is exported and can be assigned from serialized data.

Properties

- int [WordCount](#) [get, set]
Word count read from serialized data.

6.402.1 Detailed Description

Used in plugin. Indicates that NetworkBehaviour.DynamicWordCount is exported and can be assigned from serialized data.

6.402.2 Property Documentation

6.402.2.1 WordCount

```
int WordCount [get], [set]
```

Word count read from serialized data.

6.403 IPublicFacingInterface Interface Reference

Tag Interface for all public facing [Fusion](#) interfaces

Inherited by [IAfterAllTicks](#), [IAfterClientPredictionReset](#), [IAfterHostMigration](#), [IAfterRender](#), [IAfterSpawned](#), [IAfterTick](#), [IAfterUpdate](#), [IAfterUpdateRemotePrefabs](#), [IBeforeAllTicks](#), [IBeforeClientPredictionReset](#), [IBeforeCopyPreviousState](#), [IBeforeHitboxRegistration](#), [IBeforeSimulation](#), [IBeforeTick](#), [IBeforeUpdate](#), [IBeforeUpdateRemotePrefabs](#), [IDespawned](#), [IInputAuthorityGained](#), [IInputAuthorityLost](#), [IInterestEnter](#), [IInterestExit](#), [ILocalPrefabCreated](#), [INetworkRunnerCallbacks](#), [IPlayerJoined](#), [IPlayerLeft](#), [IRemotePrefabCreated](#), [ISceneLoadDone](#), [ISceneLoadStart](#), [ISimulationEnter](#), [ISimulationExit](#), [ISpawned](#), and [IStateAuthorityChanged](#).

6.403.1 Detailed Description

Tag Interface for all public facing [Fusion](#) interfaces

6.404 NetworkedWeavedStringAttribute Class Reference

An attribute emitted by the weaver to mark a string field as networked.

Inherits [Attribute](#).

Public Member Functions

- [NetworkedWeavedStringAttribute](#) (int capacity, string cacheFieldName)

Properties

- string [CacheFieldName](#) [get]
Name of the field that servers as cache
- int [Capacity](#) [get]
Max capacity of the string (in characters)

6.404.1 Detailed Description

An attribute emitted by the weaver to mark a string field as networked.

6.404.2 Constructor & Destructor Documentation

6.404.2.1 NetworkedWeavedStringAttribute()

```
NetworkedWeavedStringAttribute (
    int capacity,
    string cacheFieldName )
```

Parameters

<i>capacity</i>	Max capacity of the string (in characters)
<i>cacheFieldName</i>	Name of the field that servers as cache

6.404.3 Property Documentation

6.404.3.1 CacheFieldName

```
string CacheFieldName [get]
```

Name of the field that servers as cache

6.404.3.2 Capacity

```
int Capacity [get]
```

Max capacity of the string (in characters)

Index

- [_128, 73](#)
 - [Data, 73](#)
 - [SIZE, 73](#)
- [_16, 74](#)
 - [Data, 74](#)
 - [SIZE, 74](#)
- [_2, 75](#)
 - [Data, 75](#)
 - [SIZE, 75](#)
- [_256, 75](#)
 - [Data, 76](#)
 - [SIZE, 76](#)
- [_32, 76](#)
 - [Data, 77](#)
 - [SIZE, 77](#)
- [_4, 77](#)
 - [Data, 78](#)
 - [SIZE, 78](#)
- [_512, 78](#)
 - [Data, 79](#)
 - [SIZE, 79](#)
- [_64, 79](#)
 - [Data, 79](#)
 - [SIZE, 80](#)
- [_8, 80](#)
 - [Data, 80](#)
 - [SIZE, 80](#)
- [_debruijnTable](#)
 - [DynamicHeap, 134](#)
- [_doubleValue](#)
 - [DolfAttributeBase, 128](#)
- [_isDouble](#)
 - [DolfAttributeBase, 128](#)
- [_longValue](#)
 - [DolfAttributeBase, 128](#)
- [_padding](#)
 - [ReliableId, 1060](#)
- [_reserved](#)
 - [NetworkObjectHeader, 555](#)
- [_value](#)
 - [AtomicInt, 108](#)
- [_value0](#)
 - [NetworkObjectTypeId, 577](#)
- [_value1](#)
 - [NetworkObjectTypeId, 577](#)
- [AABB, 279](#)
 - [AABB, 280](#)
 - [Center, 281](#)
 - [Extents, 281](#)
 - [Max, 281](#)
 - [Min, 281](#)
- [Accept](#)
 - [NetworkRunnerCallbackArgs.ConnectRequest, 698](#)
- [ACCURACY](#)
 - [ReadWriteUtils, 846](#)
- [AckForceCount](#)
 - [NetConfigNotify, 1020](#)
- [AckForceTimeout](#)
 - [NetConfigNotify, 1020](#)
- [AckMaskBits](#)
 - [NetConfigNotify, 1020](#)
- [AckMaskBytes](#)
 - [NetConfigNotify, 1020](#)
- [Acquire](#)
 - [INetworkAssetSource< T >, 235](#)
- [AcquirePrefabInstance](#)
 - [INetworkObjectProvider, 239](#)
- [Active](#)
 - [NetConnection, 1027](#)
- [ActivePlayers](#)
 - [NetworkRunner, 689](#)
 - [Simulation, 915](#)
- [ActorId](#)
 - [NetAddress, 972](#)
- [Add](#)
 - [INetworkDictionary, 236](#)
 - [INetworkLinkedList, 237](#)
 - [NetworkDictionary< K, V >, 476](#)
 - [NetworkLinkedList< T >, 510](#)
 - [SerializableDictionary< TKey, TValue >, 889](#)
 - [SimulationInput.Buffer, 937](#)
- [AddAfter](#)
 - [ReliableList, 1065](#)
- [AddBefore](#)
 - [ReliableList, 1066](#)
- [AddBehaviour< T >](#)
 - [Behaviour, 111](#)
- [AddCallbacks](#)
 - [NetworkRunner, 641](#)
- [AddFirst](#)
 - [NetBitBufferList, 1000](#)
 - [ReliableList, 1066](#)
- [AddGlobal](#)
 - [NetworkRunner, 642](#)
- [AddInstance](#)
 - [NetworkPrefabTable, 603](#)
- [Additional](#)

- NetConfigSimulationOscillator, 1025
- AdditionalJitter
 - NetworkSimulationConfiguration, 725
- AdditionalLoss
 - NetworkSimulationConfiguration, 726
- AddLast
 - NetBitBufferList, 1001
 - ReliableList, 1066
- AddMode
 - NetworkRunnerUpdaterDefaultInvokeSettings, 705
- AddOnBufferTime
 - LagCompensationStatisticsSnapshot, 1086
- AddOnBVHTime
 - LagCompensationStatisticsSnapshot, 1086
- AddOnCompleted
 - NetworkSceneAsyncOp, 706
- AddPlayerAreaOfInterest
 - NetworkRunner, 642
- Address
 - NetBitBuffer, 996
 - NetConfig, 1016
 - NetPeer, 1046
 - Ptr, 832
 - StartGameArgs, 1070
- AddSceneRef
 - NetworkSceneInfo, 713
- AddSource
 - NetworkPrefabTable, 603
- AddTicks
 - TickAccumulator, 1102
- AddTime
 - TickAccumulator, 1102
- AdvanceBufferTime
 - LagCompensationStatisticsSnapshot, 1086
- After
 - Fusion, 61
- AfterAllTicks
 - IAfterAllTicks, 215
- AfterClientPredictionReset
 - IAfterClientPredictionReset, 216
- AfterHostMigration
 - IAfterHostMigration, 217
- AfterRender
 - IAfterRender, 217
- AfterSimulation
 - Simulation, 908
- AfterSpawned
 - IAfterSpawned, 218
- AfterTick
 - IAfterTick, 219
- AfterUpdate
 - IAfterUpdate, 219
 - Simulation, 908
- AfterUpdateRemotePrefabs
 - IAfterUpdateRemotePrefabs, 220
- ALIGNMENT
 - Native, 364
 - NetworkId, 499
- NetworkObjectGuid, 548
- NetworkObjectNestingKey, 564
- NetworkObjectTypeId, 578
- NetworkPrefabId, 589
- NetworkPrefabRef, 599
- Tick, 1098
- AlignPointer
 - Native, 339
- ALL
- AuthorityMasks, 109
- All
 - Fusion, 57
- Allocate
 - DynamicHeapInstance, 136
 - NetBitBuffer, 978
 - NetConnectionMap, 1032
 - SimulationMessage, 944
- AllocateArray< T >
 - DynamicHeapInstance, 136
- AllocateArrayPointers< T >
 - DynamicHeapInstance, 137
- AllocateTracked< T >
 - DynamicHeapInstance, 137
- AllocateTrackedArray< T >
 - DynamicHeapInstance, 138
- AllocateTrackedArrayPointers< T >
 - DynamicHeapInstance, 138
- Allocator, 81
 - BUCKET_COUNT, 82
 - BUCKET_INVALID, 82
 - HEAP_ALIGNMENT, 82
 - REPLICATE_WORD_ALIGN, 82
 - REPLICATE_WORD_SHIFT, 83
 - REPLICATE_WORD_SIZE, 83
- Allocator.Config, 83
 - BlockByteSize, 87
 - BlockCount, 86
 - BlockShift, 86
 - BlockWordCount, 87
 - Config, 84
 - DEFAULT_BLOCK_COUNT, 86
 - DEFAULT_BLOCK_SHIFT, 86
 - Equals, 84, 85
 - GetHashCode, 85
 - GlobalsSize, 86
 - HeapSizeAllocated, 87
 - HeapSizeUsable, 87
 - SIZE, 86
 - ToString, 85
- AllowClientServerModesInWebGL
 - NetworkProjectConfig, 615
- AllowEditMode
 - FusionGlobalScriptableObjectSourceAttribute, 186
- AllowFallback
 - FusionGlobalScriptableObjectSourceAttribute, 186
- AllowMultipleTargets
 - EditorButtonAttribute, 140
 - FieldEditorButtonAttribute, 159

- AllowStateAuthorityOverride
 - Fusion, 50
- Alpha
 - NetworkBehaviourBufferInterpolator, 433
 - Query, 296
 - QueryParams, 298
 - TickAccumulator, 1102
- AlreadyRunning
 - Fusion, 59
- Always
 - Fusion, 48
- AlwaysExpanded
 - ExpandableEnumAttribute, 158
- Angle, 90
 - Clamp, 91, 92
 - Equals, 92
 - GetHashCode, 93
 - Lerp, 93
 - Max, 93
 - Min, 93
 - NetworkBehaviourBufferInterpolator, 425
 - operator Angle, 93, 94
 - operator double, 94
 - operator float, 96
 - operator !=, 96
 - operator <, 97
 - operator <=, 98
 - operator >, 98
 - operator >=, 99
 - operator +, 96
 - operator -, 97
 - operator ==, 98
 - SIZE, 99
 - ToString, 99
- Animator
 - NetworkMecanimAnimator, 526
- Any
 - NetAddress, 969
- AnyIPv6
 - NetAddress, 970
- ApplyTiming
 - NetworkMecanimAnimator, 526
- AreaOfInterest
 - Fusion, 50
- AreaOfInterestEnabled
 - SimulationConfig, 933
- AreaOfInterestOverride
 - NetworkTRSPData, 766
- ArrayClear< T >
 - Native, 339
- ArrayCompare< T >
 - Native, 340
- ArrayCopy
 - Native, 340
- ArrayElementSerializer< T >
 - BitStream, 788
- ArrayLengthAttribute, 100
 - ArrayLengthAttribute, 100, 101
 - MaxLength, 101
 - MinLength, 101
- AsBox
 - ErrorIfAttribute, 157
 - WarnIfAttribute, 1158
- AsCustom
 - NetworkObjectTypeId, 578
- AsIndex
 - NetworkPrefabId, 590
 - PlayerRef, 777
 - SceneRef, 886
- AsInternalStructId
 - NetworkObjectTypeId, 579
- AsPathHash
 - SceneRef, 886
- AspectRatio
 - NormalizedRectAttribute, 769
- AsPrefabId
 - NetworkObjectTypeId, 579
- AsSceneObjectId
 - NetworkObjectTypeId, 579
- AssembliesToWeave
 - NetworkProjectConfig, 615
- AssemblyFileVersion
 - Versioning, 1156
- AssemblyNameAttribute, 101
 - RequiresUnsafeCode, 102
- AssemblyQualifiedName
 - SerializableType< BaseType >, 898
- AssetGuid
 - INetworkPrefabSource, 241
- AssetObject, 102
- AsShort
 - SerializableType< BaseType >, 897
- Assign
 - NetworkString< TSize >, 735
- AssignInputAuthority
 - NetworkObject, 529
- AtomicInt, 106
 - _value, 108
 - AtomicInt, 107
 - CompareExchange, 107
 - Decrement, 107
 - Exchange, 108
 - IncrementPost, 108
 - IncrementPre, 108
 - Value, 109
- Attach
 - NetworkRunner, 642, 643
- AuthenticationTicketExpired
 - Fusion, 59
- AuthenticationValues
 - NetworkRunner, 689
- AuthorityMasks, 109
 - ALL, 109
 - INPUT, 110
 - NONE, 110
 - PROXY, 110

- STATE, 110
- AuthValues
 - StartGameArgs, 1070
- AutoHostOrClient
 - Fusion, 48
- AutoUpdateAreaOfInterestOverride
 - NetworkTransform, 762
- Available
 - TickRate, 1111
- Awaiter
 - NetworkSceneAsyncOp.Awaiter, 710
 - NetworkSpawnOp.Awaiter, 730
- Awake
 - NetworkObject, 529
- BackColor
 - ScriptHelpAttribute, 887
- BaseType
 - SerializableTypeAttribute, 899
- Before
 - Fusion, 61
- BeforeAllTicks
 - IBeforeAllTicks, 222
- BeforeClientPredictionReset
 - IBeforeClientPredictionReset, 222
- BeforeCopyPreviousState
 - IBeforeCopyPreviousState, 223
- BeforeFirstTick
 - Simulation, 908
- BeforeHitboxRegistration
 - IBeforeHitboxRegistration, 224
- BeforeSimulation
 - IBeforeSimulation, 224
 - Simulation, 908
- BeforeTick
 - IBeforeTick, 225
- BeforeUpdate
 - IBeforeUpdate, 226
 - Simulation, 908
- BeforeUpdateRemotePrefabs
 - IBeforeUpdateRemotePrefabs, 226
- Begin
 - EngineProfiler, 148
- Behaviour, 110
 - AddBehaviour< T >, 111
 - DestroyBehaviour, 111
 - GetBehaviour< T >, 111
 - NetworkBehaviourBufferInterpolator, 433
 - NetworkBehaviourId, 437
 - RpcHeader, 866
 - TryGetBehaviour< T >, 112
- BehaviourCount
 - NetworkObjectHeader, 555
- BehaviourMeta
 - NetworkProjectConfigAsset, 622
- BehaviourStatisticsManager, 1077
 - CompletedSnapshot, 1077
- BehaviourStatisticsSnapshot, 1078
 - FixedUpdateNetworkExecutionCount, 1078
 - FixedUpdateNetworkExecutionTime, 1078
 - RenderExecutionCount, 1078
 - RenderExecutionTime, 1078
- BinaryDataAttribute, 112
- Bind
 - INetSocket, 964
- BinUtils, 112
 - BytesToHex, 113, 114
 - ByteToHex, 114
 - HexToBytes, 114
 - unsafe, 115
 - WordsToHex, 115, 116
- BitCount
 - BitSetAttribute, 117
- Bits
 - NetworkButtons, 468
- BitScanReverse
 - Maths, 318–320
- BitSetAttribute, 116
 - BitCount, 117
 - BitSetAttribute, 117
- BitsRequiredForNumber
 - Maths, 320
- BitStream, 782
 - ArrayElementSerializer< T >, 788
 - BitStream, 787, 788
 - BytesRequired, 822
 - CanRead, 789
 - CanWrite, 789
 - Capacity, 823
 - Condition, 790
 - CopyFromArray, 790
 - Data, 823
 - Done, 823
 - Expand, 790
 - IsEvenBytes, 823
 - Overflowing, 823
 - Position, 823
 - ReadBool, 790
 - ReadBoolean, 791
 - ReadByte, 791
 - ReadByteArray, 792, 793
 - ReadByteArrayLengthPrefixed, 793
 - ReadChar, 793
 - ReadDouble, 793
 - ReadFloat, 794
 - ReadGuid, 794
 - Reading, 824
 - ReadInt, 794
 - ReadInt_Shifted, 795
 - ReadLong, 795
 - ReadSByte, 796
 - ReadShort, 796
 - ReadString, 797
 - ReadUInt, 797
 - ReadULong, 798
 - ReadUShort, 798
 - Reset, 799

- ResetFast, [799](#)
- RoundToByte, [800](#)
- Serialize, [800–804](#), [806–808](#)
- SerializeArray< T >, [808](#)
- SerializeArrayLength< T >, [809](#)
- SerializeBuffer, [809–811](#)
- SetBuffer, [812](#)
- Size, [824](#)
- ToArray, [812](#)
- WriteBool, [812](#)
- WriteBoolean, [813](#)
- WriteByte, [813](#)
- WriteByteArray, [814](#)
- WriteByteArrayLengthPrefixed, [815](#)
- WriteByteAt, [815](#)
- WriteChar, [817](#)
- WriteDouble, [817](#)
- WriteFloat, [817](#)
- WriteGuid, [817](#)
- WriteInt, [818](#)
- WriteInt_Shifted, [818](#)
- WriteLong, [819](#)
- WriteSByte, [819](#)
- WriteShort, [820](#)
- WriteString, [820](#)
- WriteUInt, [821](#)
- WriteULong, [821](#), [822](#)
- WriteUShort, [822](#)
- Writing, [824](#)
- BitwiseAndNotEqualZero
 - Fusion, [47](#)
- BLOCK_SIZE
 - NetworkId, [499](#)
- BlockByteSize
 - Allocator.Config, [87](#)
- BlockCount
 - Allocator.Config, [86](#)
- BlockShift
 - Allocator.Config, [86](#)
- BlockWordCount
 - Allocator.Config, [87](#)
- Bool
 - NetworkBehaviourBufferInterpolator, [425](#), [426](#)
- Bounds
 - BVHNodeDrawInfo, [287](#)
- Box
 - Fusion, [49](#)
- Box2D
 - Fusion.LagCompensation, [65](#)
- BoxExtents
 - ColliderDrawInfo, [288](#)
 - Hitbox, [191](#)
- BoxOverlapQuery, [281](#)
 - BoxOverlapQuery, [282](#)
 - Center, [283](#)
 - Check, [283](#)
 - Extents, [283](#)
 - Rotation, [284](#)
- BoxOverlapQueryParams, [284](#)
 - BoxOverlapQueryParams, [284](#)
 - Center, [285](#)
 - Extents, [285](#)
 - QueryParams, [285](#)
 - Rotation, [285](#)
 - StaticHitsCapacity, [285](#)
- BroadRadius
 - HitboxRoot, [211](#)
- BUCKET_COUNT
 - Allocator, [82](#)
 - MemoryStatisticsSnapshot, [1088](#)
- BUCKET_INVALID
 - Allocator, [82](#)
- BucketFreeBlocksCount
 - MemoryStatisticsSnapshot, [1088](#)
- BucketFullBlocksCount
 - MemoryStatisticsSnapshot, [1089](#)
- BucketUsedBlocksCount
 - MemoryStatisticsSnapshot, [1089](#)
- Buffer
 - NetBitBufferSerializer, [1010](#)
 - SimulationInput.Buffer, [936](#)
- BuildType
 - NetworkRunner, [689](#)
- BuildTypes
 - NetworkProjectConfig, [615](#)
 - NetworkRunner, [641](#)
- BVHDepth
 - HitboxManager, [206](#)
- BVHDraw, [286](#)
 - GetEnumerator, [286](#)
 - LagCompensationDraw, [292](#)
- BVHMaxDeep
 - LagCompensationStatisticsSnapshot, [1086](#)
- BVHNodeDrawInfo, [286](#)
 - Bounds, [287](#)
 - Depth, [287](#)
 - MaxDepth, [287](#)
- BVHNodes
 - HitboxManager, [207](#)
- BVHNodesCount
 - LagCompensationStatisticsSnapshot, [1086](#)
- ByRemote
 - Fusion.Sockets, [69](#)
- ByteCount
 - MaxStringByteCountAttribute, [336](#)
 - NetworkObjectHeader, [557](#)
- Bytes
 - Fusion, [60](#)
- BytesRemaining
 - NetBitBuffer, [997](#)
- BytesRequired
 - BitStream, [822](#)
- BytesRequiredForBits
 - Maths, [321](#)
- BytesToHex
 - BinUtils, [113](#), [114](#)

- ByteToHex
 - BinUtils, 114
- CACHE_LINE_SIZE
 - Native, 364
- CachedStaticCollidersSize
 - LagCompensationSettings, 310
- CacheFieldName
 - NetworkedWeavedStringAttribute, 1161
- CanAllocateUserPayload
 - SimulationMessage, 944
- CanFragment
 - INetSocket, 965
- CanRead
 - BitStream, 789
 - NetBitBuffer, 979
- CanReceiveRenderCallback
 - SimulationBehaviour, 927
- CanReceiveSimulationCallback
 - SimulationBehaviour, 927
- CanSpawn
 - NetworkRunner, 689
- CanWrite
 - BitStream, 789
 - NetBitBuffer, 979
- Capacity
 - BitStream, 823
 - FixedBufferPropertyAttribute, 173
 - NetworkDictionary< K, V >, 479
 - NetworkDictionaryReadOnly< K, V >, 483
 - NetworkedWeavedArrayAttribute, 485
 - NetworkedWeavedDictionaryAttribute, 488
 - NetworkedWeavedLinkedListAttribute, 489
 - NetworkedWeavedStringAttribute, 1161
 - NetworkLinkedList< T >, 513
 - NetworkLinkedListReadOnly< T >, 519
 - NetworkString< TSize >, 757
 - SimulationMessage, 949
- CapacityAttribute, 117
 - CapacityAttribute, 118
 - Length, 118
- Capsule
 - Fusion, 49
- CapsuleBottomCenter
 - ColliderDrawInfo, 288
- CapsuleExtents
 - ColliderDrawInfo, 288
 - Hitbox, 191
- CapsuleRadius
 - Hitbox, 192
- CapsuleTopCenter
 - ColliderDrawInfo, 288
- CeilToInt
 - Maths, 321
- CELL_SIZE
 - Simulation.AreaOfInterest, 924
- Center
 - AABB, 281
 - BoxOverlapQuery, 283
 - BoxOverlapQueryParams, 285
 - SphereOverlapQuery, 307
 - SphereOverlapQueryParams, 308
- ChangeConnectionAddressDuringConnecting
 - NetPeerGroup, 1049
- Changed
 - NetworkBehaviour.ChangeDetector.Enumerable, 410
- ChangedTick
 - NetworkBehaviour, 403
- Channel
 - RpcAttribute, 862
 - RpcInfo, 869
- CharacterCount
 - UTF32Tools.ConversionResult, 1140
- Check
 - BoxOverlapQuery, 283
 - NetBitBufferSerializer, 1007
 - Query, 296
 - RaycastQuery, 302
 - SphereOverlapQuery, 307
- Check_ENABLE_IL2CPP
 - RuntimeUnityFlagsSetup, 875
- Check_ENABLE_MONO
 - RuntimeUnityFlagsSetup, 875
- Check_NET_4_6
 - RuntimeUnityFlagsSetup, 876
- Check_NET_STANDARD_2_0
 - RuntimeUnityFlagsSetup, 876
- Check_NETFX_CORE
 - RuntimeUnityFlagsSetup, 876
- Check_UNITY_2019_4_OR_NEWER
 - RuntimeUnityFlagsSetup, 876
- Check_UNITY_EDITOR
 - RuntimeUnityFlagsSetup, 876
- Check_UNITY_GAMECORE
 - RuntimeUnityFlagsSetup, 876
- Check_UNITY_SWITCH
 - RuntimeUnityFlagsSetup, 877
- Check_UNITY_WEBGL
 - RuntimeUnityFlagsSetup, 877
- Check_UNITY_XBOXONE
 - RuntimeUnityFlagsSetup, 877
- CheckBitCount
 - NetBitBuffer, 979
- CheckNetworkedPropertiesBeingEmpty
 - NetworkProjectConfig, 615
- CheckRpcAttributeUsage
 - NetworkProjectConfig, 616
- Clamp
 - Angle, 91, 92
 - Maths, 322, 323
- Clamp01
 - Maths, 323, 324
- ClampMax
 - RangeExAttribute, 840
- ClampMin
 - RangeExAttribute, 840

- ClampSelection
 - TickRate, [1107](#)
- ClampToByte
 - Maths, [324](#)
- Clear
 - FixedArray< T >, [165](#)
 - Mask256, [314](#)
 - NetBitBuffer, [980](#)
 - NetworkArray< T >, [374](#)
 - NetworkDictionary< K, V >, [476](#)
 - NetworkLinkedList< T >, [510](#)
 - NetworkPrefabTable, [603](#)
 - SerializableDictionary< TKey, TValue >, [891](#)
 - SimulationInput, [934](#)
 - SimulationInput.Buffer, [937](#)
- ClearMonitoredNetworkObjects
 - NetworkObjectStatisticsManager, [1090](#)
- ClearPlayerAreaOfInterest
 - NetworkRunner, [643](#)
- Client
 - Fusion, [48](#), [59](#)
 - TickRate, [1111](#)
 - TickRate.Resolved, [1113](#)
 - TickRate.Selection, [1115](#)
- ClientSend
 - TickRate.Resolved, [1113](#)
- ClientSendDelta
 - TickRate.Resolved, [1114](#)
- ClientSendIndex
 - TickRate.Selection, [1115](#)
- ClientSendIndexOutOfRange
 - TickRate, [1106](#)
- ClientServer
 - Fusion, [58](#), [60](#)
- ClientTickDelta
 - TickRate.Resolved, [1114](#)
- ClientTickStride
 - TickRate.Resolved, [1114](#)
- ClientToClientWithServerProxy
 - NetworkConfiguration, [470](#)
- ClientToServer
 - NetworkConfiguration, [470](#)
- Clone
 - NetworkSimulationConfiguration, [725](#)
 - SimulationMessage, [944](#)
- CloneArray< T >
 - NetworkBehaviourUtils, [440](#)
- CloudConnectionLostHandler
 - NetworkRunner, [643](#)
- CodePointCount
 - UTF32Tools.ConversionResult, [1140](#)
- CollectGarbage
 - DynamicHeap, [133](#)
- CollectGarbageDelegate
 - DynamicHeap, [133](#)
- Collider
 - LagCompensatedHit, [277](#)
- Collider2D
 - LagCompensatedHit, [278](#)
- ColliderDrawInfo, [287](#)
- BoxExtents, [288](#)
- CapsuleBottomCenter, [288](#)
- CapsuleExtents, [288](#)
- CapsuleTopCenter, [288](#)
- LocalToWorldMatrix, [288](#)
- Offset, [289](#)
- Radius, [289](#)
- Type, [289](#)
- ColliderIndex
 - Hitbox, [193](#)
- CommunicatorID
 - ICommunicator, [827](#)
- Compare
 - DolfAttributeBase, [128](#)
 - NetworkObjectSortKeyComparer, [570](#)
 - NetworkString< TSize >, [735](#), [736](#)
 - Tick.RelationalComparer, [1100](#)
- Compare< TOtherSize >
 - NetworkString< TSize >, [736](#), [737](#)
- CompareExchange
 - AtomicInt, [107](#)
- CompareOperator
 - Fusion, [47](#)
- Comparer
 - NetworkId, [500](#)
 - NetworkObjectTypeId, [579](#)
 - PlayerRef, [777](#)
- CompareTo
 - NetworkId, [496](#)
 - NetworkObjectGuid, [544](#)
 - NetworkPrefabId, [587](#)
 - NetworkPrefabRef, [595](#)
 - Tick, [1093](#)
- Completed
 - IAsyncOperation, [221](#)
- CompletedSnapshot
 - BehaviourStatisticsManager, [1077](#)
- CompleteSnapshot
 - FusionStatisticsManager, [1079](#)
- Compress
 - FloatUtils, [178](#)
- Compute
 - CRC64, [118](#), [120](#)
- ComputeHash
 - IDataEncryption, [143](#)
- Condition
 - BitStream, [790](#)
- ConditionMember
 - DolfAttributeBase, [128](#)
- Config
 - Allocator.Config, [84](#)
 - HitboxRoot, [211](#)
 - NetPeer, [1046](#)
 - NetworkProjectConfigAsset, [622](#)
 - NetworkRunner, [690](#)
 - Simulation, [915](#)

- StartGameArgs, [1071](#)
- ConfigFlags
 - HitboxRoot, [209](#)
- Connect
 - NetPeerGroup, [1049](#)
- ConnectAttempts
 - NetConfig, [1016](#)
 - NetworkConfiguration, [471](#)
- Connected
 - Fusion.Sockets, [69](#)
- Connecting
 - Fusion.Sockets, [69](#)
- ConnectInterval
 - NetConfig, [1016](#)
 - NetworkConfiguration, [471](#)
- ConnectionCount
 - NetPeerGroup, [1055](#)
- ConnectionDefaultRtt
 - NetConfig, [1016](#)
 - NetworkConfiguration, [471](#)
- ConnectionGroups
 - NetConfig, [1016](#)
- ConnectionIterator
 - NetPeerGroup, [1050](#)
- ConnectionPingInterval
 - NetConfig, [1016](#)
 - NetworkConfiguration, [471](#)
- ConnectionRefused
 - Fusion, [59](#)
- ConnectionsBuffer
 - NetConnectionMap, [1036](#)
- ConnectionSendBuffers
 - NetConfig, [1017](#)
- ConnectionShutdownTime
 - NetConfig, [1017](#)
 - NetworkConfiguration, [470](#)
- ConnectionsPerGroup
 - NetConfig, [1018](#)
- ConnectionStatus
 - NetConnection, [1027](#)
- ConnectionTimeout
 - Fusion, [59](#)
 - NetConfig, [1017](#)
 - NetworkConfiguration, [470](#)
- ConnectionToken
 - StartGameArgs, [1071](#)
- ConnectionType
 - Fusion, [47](#)
- ConsumeTick
 - TickAccumulator, [1103](#)
- Contains
 - NetworkLinkedList< T >, [511](#)
 - NetworkLinkedListReadOnly< T >, [518](#)
 - NetworkPrefabTable, [604](#)
 - NetworkString< TSize >, [737](#), [738](#)
 - SimulationInput.Buffer, [937](#)
- Contains< TOtherSize >
 - NetworkString< TSize >, [739](#)
- ContainsKey
 - NetworkDictionary< K, V >, [476](#)
 - SerializableDictionary< TKey, TValue >, [891](#)
- ContainsValue
 - NetworkDictionary< K, V >, [476](#)
- ConterMask
 - Fusion, [51](#)
- ContinueWhenAll
 - TaskManager, [103](#)
- ConversionResult
 - UTF32Tools.ConversionResult, [1140](#)
- Convert
 - UTF32Tools, [1136](#), [1137](#)
- Convert< T >
 - NetworkInput, [502](#)
- CopyBackingFieldsToState
 - NetworkBehaviour, [387](#)
- CopyFrom
 - FixedArray< T >, [165](#)
 - NetworkArray< T >, [375](#)
 - SimulationInput, [935](#)
- CopyFromArray
 - BitStream, [790](#)
- CopyFromArray< T >
 - Native, [341](#)
- CopyFromNetworkArray< T >
 - NetworkBehaviourUtils, [440](#)
- CopyFromNetworkDictionary< D, K, V >
 - NetworkBehaviourUtils, [441](#)
- CopyFromNetworkList< T >
 - NetworkBehaviourUtils, [441](#)
- CopySortedTo
 - SimulationInput.Buffer, [938](#)
- CopyStateFrom
 - NetworkBehaviour, [387](#)
 - NetworkObject, [529](#), [530](#)
- CopyStateToBackingFields
 - NetworkBehaviour, [387](#)
- CopyTo
 - FixedArray< T >, [166](#)
 - NetworkArray< T >, [376](#)
- CopyToArray< T >
 - Native, [341](#)
- CosineInterpolate
 - Maths, [325](#)
- Count
 - Fusion, [60](#)
 - NetConnectionMap, [1036](#)
 - NetworkDictionary< K, V >, [479](#)
 - NetworkDictionaryReadOnly< K, V >, [483](#)
 - NetworkLinkedList< T >, [513](#)
 - NetworkLinkedListReadOnly< T >, [519](#)
 - ReliableList, [1067](#)
 - SerializableDictionary< TKey, TValue >, [894](#)
 - SimulationInput.Buffer, [939](#)
 - TickRate, [1111](#)
- CountSetBits
 - Maths, [325](#)

- CountUsed
 - NetConnectionMap, 1036
- CountUsedBitsMinOne
 - Maths, 325
- CRC64, 118
 - Compute, 118, 120
- Create
 - INetSocket, 965
 - NetCommandHeader, 1013
 - NetworkSimulationConfiguration, 725
 - RpcHeader, 864, 865
- Create< T >
 - FixedArray< T >, 166
- Create< TActual, TAdapted >
 - FixedArray< T >, 167
- Create< TKey, TValue >
 - SerializableDictionary< TKey, TValue >, 891
- Created
 - Fusion.Sockets, 69
- CreateFromFieldSequence< T >
 - FixedArray< T >, 167
- CreateFromIpPort
 - NetAddress, 970
- CreateFromSeconds
 - TickTimer, 1117
- CreateFromTicks
 - TickTimer, 1117
- Current
 - FixedArray< T >.Enumerator, 171
 - NetConnectionMap.Iterator, 1038
 - NetworkArray< T >.Enumerator, 380
 - NetworkBehaviour.ChangeDetector.Enumerator, 412
 - NetworkDictionary< K, V >.Enumerator, 481
 - NetworkLinkedList< T >.Enumerator, 515
 - UTF32Tools.CharEnumerator, 1139
- CurrentConnectionType
 - NetworkRunner, 690
- CurrentTypeId
 - NetworkProjectConfig, 616
- CurrentVersion
 - NetworkProjectConfig, 616
- Custom
 - Fusion, 52, 58
- CustomAuthenticationFailed
 - Fusion, 59
- CustomCallbackInterfaces
 - StartGameArgs, 1071
- CustomLobbyName
 - StartGameArgs, 1071
- CustomPhotonAppSettings
 - StartGameArgs, 1071
- CustomPublicAddress
 - StartGameArgs, 1071
- CustomSTUNServer
 - StartGameArgs, 1072
- Data
 - _128, 73
 - _16, 74
 - _2, 75
 - _256, 76
 - _32, 77
 - _4, 78
 - _512, 79
 - _64, 79
 - _8, 80
 - BitStream, 823
 - NetBitBuffer, 997
 - NetworkInput, 504
 - NetworkPrefabAcquireContext, 584
 - NetworkPrefabInfo, 592
 - NetworkTRSP, 765
 - ReliableKey, 1064
 - SimulationInput, 935
- DataConsistency
 - SimulationConfig, 930
- DataEncryptor, 141
 - Dispose, 142
 - EncryptData, 142
- DataProperty
 - IUnityValueSurrogate< T >, 257
 - UnityArraySurrogate< T, ReaderWriter >, 259
 - UnityDictionarySurrogate< TKeyType, TKeyReaderWriter, TValueType, TValueReaderWriter >, 261
 - UnityLinkedListSurrogate< T, ReaderWriter >, 263
 - UnityValueSurrogate< T, TReaderWriter >, 267
- Debug
 - NetworkRunner, 641
- Decompress
 - FloatUtils, 178
- DecoratingPropertyAttribute, 120
 - DecoratingPropertyAttribute, 121
 - DefaultOrder, 121
- Decrement
 - AtomicInt, 107
- DecryptData
 - IDataEncryption, 144
- Default
 - HitboxRoot, 209
 - NetworkedAttribute, 484
 - NetworkPrefabTableOptions, 610
- DEFAULT_ACCURACY
 - FloatUtils, 179
- DEFAULT_BLOCK_COUNT
 - Allocator.Config, 86
- DEFAULT_BLOCK_SHIFT
 - Allocator.Config, 86
- DEFAULT_HEADERS
 - NetPeer, 1045
- DefaultContents
 - FusionGlobalScriptableObjectAttribute, 183
- DefaultContentsGeneratorMethod
 - FusionGlobalScriptableObjectAttribute, 183
- DefaultForPropertyAttribute, 122
 - DefaultForPropertyAttribute, 122

- PropertyName, [123](#)
- WordCount, [123](#)
- WordOffset, [123](#)
- DefaultOrder
 - DecoratingPropertyAttribute, [121](#)
- DefaultPath
 - FusionGlobalScriptableObjectAttribute, [183](#)
- DefaultResourceName
 - NetworkProjectConfig, [616](#)
- Defaults
 - NetConfig, [1018](#)
 - NetConfigNotify, [1021](#)
 - NetConfigSimulation, [1023](#)
- Degrees
 - Fusion, [60](#)
- DegreesPerSecond
 - Fusion, [60](#)
- Delay
 - TaskManager, [103](#)
- DelayMax
 - NetworkSimulationConfiguration, [726](#)
- DelayMin
 - NetworkSimulationConfiguration, [726](#)
- DelayOscillator
 - NetConfigSimulation, [1022](#)
- DelayPeriod
 - NetworkSimulationConfiguration, [726](#)
- DelayShape
 - NetworkSimulationConfiguration, [726](#)
- DelayThreshold
 - NetworkSimulationConfiguration, [726](#)
- Delegate
 - RpclInvokeData, [871](#)
- DeleteEncryptionKey
 - INetSocket, [965](#)
- DeltaTime
 - NetworkRunner, [690](#)
 - Simulation, [915](#)
 - SimulationConfig, [931](#)
- Depth
 - BVHNodeDrawInfo, [287](#)
- Description
 - INetworkAssetSource< T >, [235](#)
- Deserialize
 - NetworkProjectConfig, [613](#)
- Despawn
 - NetworkRunner, [643](#)
- Despawned
 - IDespawned, [227](#)
 - NetworkBehaviour, [387](#)
- Destroy
 - INetSocket, [966](#)
 - NetPeer, [1040](#)
- DestroyBehaviour
 - Behaviour, [111](#)
- DestroySingleton< T >
 - NetworkRunner, [644](#)
- DestroyWhenStateAuthorityLeaves
 - Fusion, [50](#)
- DetectChanges
 - NetworkBehaviour.ChangeDetector, [408](#), [409](#)
- Direct
 - Fusion, [47](#)
- Direction
 - RaycastQuery, [302](#)
 - RaycastQueryParams, [304](#)
- DirtyObject
 - EditorButtonAttribute, [141](#)
- DisableNATPunchthrough
 - StartGameArgs, [1072](#)
- DisableSharedModelInterpolation
 - NetworkTransform, [761](#)
- Disconnect
 - NetPeerGroup, [1050](#)
 - NetworkRunner, [644](#)
- Disconnected
 - Fusion.Sockets, [69](#)
- DisconnectedByPluginLogic
 - Fusion, [59](#)
- DisconnectReason
 - Fusion.Protocol, [66](#)
- DisplayAsEnumAttribute, [123](#)
 - DisplayAsEnumAttribute, [124](#)
 - EnumType, [124](#)
 - EnumTypeMemberName, [124](#)
- DisplayNameAttribute, [125](#)
 - DisplayNameAttribute, [125](#)
 - Name, [126](#)
- Dispose
 - DataEncryptor, [142](#)
 - FixedArray< T >.Enumerator, [171](#)
 - NetConnectionMap, [1033](#)
 - NetworkArray< T >.Enumerator, [380](#)
 - NetworkDictionary< K, V >.Enumerator, [480](#)
 - NetworkLinkedList< T >.Enumerator, [514](#)
 - SimulationBehaviourListScope, [929](#)
 - UTF32Tools.CharEnumerator, [1138](#)
- Distance
 - LagCompensatedHit, [278](#)
- DolfAttributeBase, [126](#)
 - _doubleValue, [128](#)
 - _isDouble, [128](#)
 - _longValue, [128](#)
 - Compare, [128](#)
 - ConditionMember, [128](#)
 - DolfAttributeBase, [127](#)
 - ErrorOnConditionMemberNotFound, [129](#)
- Done
 - BitStream, [823](#)
 - NetBitBuffer, [997](#)
- DoneOrOverflow
 - NetBitBuffer, [997](#)
- DontDestroyOnLoad
 - Fusion, [50](#), [51](#)
 - NetworkPrefabAcquireContext, [584](#)
- DoubleArray< T >

- Native, [342](#)
- DoublePtrArray< T >
 - Native, [343](#)
- DrawerPropertyAttribute, [129](#)
- DrawGizmos
 - Hitbox, [191](#)
 - HitboxRoot, [209](#)
- DrawIfAttribute, [129](#)
 - DrawIfAttribute, [130](#)
 - Hide, [131](#)
 - Mode, [130](#)
- DrawIfMode
 - Fusion, [47](#)
- DrawInfo
 - HitboxManager, [207](#)
- DrawInlineAttribute, [131](#)
- DuplicateChance
 - NetConfigSimulation, [1022](#)
- DynamicHeap, [131](#)
 - _debruijnTable, [134](#)
 - CollectGarbage, [133](#)
 - CollectGarbageDelegate, [133](#)
 - Free, [133](#)
 - SetForcedAlive< T >, [134](#)
- DynamicHeap.Ignore, [135](#)
- DynamicHeap.Instance, [135](#)
 - Allocate, [136](#)
 - AllocateArray< T >, [136](#)
 - AllocateArrayPointers< T >, [137](#)
 - AllocateTracked< T >, [137](#)
 - AllocateTrackedArray< T >, [138](#)
 - AllocateTrackedArrayPointers< T >, [138](#)
 - DynamicHeap.Instance, [135](#)
 - Free, [139](#)
- DynamicWordCount
 - NetworkBehaviour, [404](#)
 - NetworkMecanimAnimator, [526](#)
- EditMode
 - Fusion, [48](#)
- EditorButtonAttribute, [139](#)
 - AllowMultipleTargets, [140](#)
 - DirtyObject, [141](#)
 - EditorButtonAttribute, [140](#)
 - Label, [141](#)
 - Priority, [141](#)
 - Visibility, [141](#)
- EditorButtonVisibility
 - Fusion, [48](#)
- ElapsedInMilliseconds
 - Timer, [1122](#)
- ElapsedInSeconds
 - Timer, [1122](#)
- ElapsedInTicks
 - Timer, [1122](#)
- ELEMENT_WORDS
 - NetworkLinkedList< T >, [513](#)
 - NetworkLinkedListReadOnly< T >, [519](#)
- ElementReaderWriterType
 - NetworkedWeavedArrayAttribute, [485](#)
 - NetworkedWeavedLinkedListAttribute, [489](#)
- ElementWordCount
 - NetworkedWeavedArrayAttribute, [486](#)
 - NetworkedWeavedLinkedListAttribute, [489](#)
- Empty
 - NetworkObjectGuid, [549](#)
 - NetworkPrefabRef, [600](#)
- Empty< T >
 - Native, [343](#)
- EnableAutoUpdate
 - HostMigrationConfig, [213](#)
- EnableClientSessionCreation
 - StartGameArgs, [1072](#)
- Enabled
 - LagCompensationSettings, [310](#)
 - NetworkSimulationConfiguration, [727](#)
- EnableEncryption
 - EncryptionConfig, [146](#)
- Encoding
 - MaxStringByteCountAttribute, [336](#)
- EncryptData
 - DataEncryptor, [142](#)
 - IDataEncryption, [144](#)
- EncryptionConfig, [146](#)
 - EnableEncryption, [146](#)
 - NetworkProjectConfig, [616](#)
- End
 - EngineProfiler, [149](#)
- EndsWith
 - NetworkString< TSize >, [740](#)
- EndsWith< TOtherSize >
 - NetworkString< TSize >, [740](#)
- EngineProfiler, [146](#)
 - Begin, [148](#)
 - End, [149](#)
 - InputQueue, [149](#)
 - InputQueueCallback, [153](#)
 - InputRecvDelta, [149](#)
 - InputRecvDeltaCallback, [154](#)
 - InputRecvDeltaDeviation, [149](#)
 - InputRecvDeltaDeviationCallback, [154](#)
 - InputSize, [150](#)
 - InputSizeCallback, [154](#)
 - InterpolationOffset, [150](#)
 - InterpolationOffsetCallback, [154](#)
 - InterpolationOffsetDeviation, [150](#)
 - InterpolationOffsetDeviationCallback, [154](#)
 - InterpolationSpeed, [150](#)
 - InterpolationSpeedCallback, [154](#)
 - Resimulations, [151](#)
 - ResimulationsCallback, [155](#)
 - RoundTripTime, [151](#)
 - RoundTripTimeCallback, [155](#)
 - RpcIn, [151](#)
 - RpcInCallback, [155](#)
 - RpcOut, [151](#)
 - RpcOutCallback, [155](#)

- SimulationOffset, 152
- SimulationOffsetCallback, 155
- SimulationOffsetDeviation, 152
- SimulationOffsetDeviationCallback, 155
- SimulationSpeed, 152
- SimulationSpeedCallback, 156
- StateRecvDelta, 153
- StateRecvDeltaCallback, 156
- StateRecvDeltaDeviation, 153
- StateRecvDeltaDeviationCallback, 156
- WorldSnapshotSize, 153
- WorldSnapshotSizeCallback, 156
- EnqueueIncompleteSynchronousSpawns
 - NetworkProjectConfig, 616
- EnsureRootComponentExists< T, TStopOn >
 - NestedComponentUtilities, 366
- EnsureRunnerScenelsActive
 - NetworkRunner, 644
- EntryKeyPropertyPath
 - SerializableDictionary< TKey, TValue >, 893
- EntryState
 - NetConnectionMap, 1032
- Enumerator
 - FixedArray< T >.Enumerator, 170
 - NetworkArray< T >.Enumerator, 379
- EnumType
 - DisplayAsEnumAttribute, 124
- EnumTypeMemberName
 - DisplayAsEnumAttribute, 124
- Equal
 - Fusion, 47
- Equals
 - Allocator.Config, 84, 85
 - Angle, 92
 - FloatCompressed, 175
 - Mask256, 314
 - NetAddress, 971
 - NetAddress.EqualityComparer, 974
 - NetConnectionId, 1029
 - NetConnectionId.EqualityComparer, 1030
 - NetworkBehaviourId, 435
 - NetworkBool, 457
 - NetworkButtons, 461
 - NetworkId, 497
 - NetworkId.EqualityComparer, 501
 - NetworkLoadSceneParameters, 521
 - NetworkObjectGuid, 545
 - NetworkObjectGuid.EqualityComparer, 550
 - NetworkObjectHeader, 552
 - NetworkObjectNestingKey, 563
 - NetworkObjectNestingKey.EqualityComparer, 565
 - NetworkObjectTypeId, 574
 - NetworkObjectTypeId.EqualityComparer, 581
 - NetworkPrefabId, 587
 - NetworkPrefabId.EqualityComparer, 591
 - NetworkPrefabRef, 596
 - NetworkPrefabRef.EqualityComparer, 601
 - NetworkRunnerUpdaterDefaultInvokeSettings, 703
 - NetworkSceneInfo, 713
 - NetworkSceneLoadId, 718
 - NetworkSceneObjectId, 721, 722
 - NetworkString< TSize >, 741, 742
 - PlayerRef, 772, 773
 - Ptr, 829
 - Ptr.EqualityComparer, 833
 - QuaternionCompressed, 835
 - SceneRef, 881
 - SerializableType< BaseType >, 897
 - Tick, 1093, 1094
 - Tick.EqualityComparer, 1099
 - Vector2Compressed, 1141, 1142
 - Vector3Compressed, 1146
 - Vector4Compressed, 1151, 1152
- Equals< TOtherSize >
 - NetworkString< TSize >, 742, 743
- Error
 - Fusion, 58
 - Fusion.Protocol, 66
 - IAsyncOperation, 221
 - NetworkSceneAsyncOp, 709
 - TickRate, 1106
- ErrorIfAttribute, 156
 - AsBox, 157
 - Message, 157
- ErrorMessage
 - StartGameResult, 1076
- ErrorOnConditionMemberNotFound
 - DolfAttributeBase, 129
- Eventual
 - SimulationConfig, 931
- Exchange
 - AtomicInt, 108
- ExecutionOrder
 - NetworkProjectConfigAsset.SerializableSimulationBehaviourMeta, 623
- Exists
 - NetworkRunner, 645
- Expand
 - BitStream, 790
 - Native, 343
- ExpandableEnumAttribute, 157
 - AlwaysExpanded, 158
 - ShowFlagsButtons, 158
 - ShowInlineHelp, 158
- ExpandArray< T >
 - Native, 344
- ExpandPtrArray< T >
 - Native, 344
- ExpansionFactor
 - LagCompensationSettings, 311
- Expired
 - TickTimer, 1118
- ExpiredOrNotRunning
 - TickTimer, 1118
- Extents
 - AABB, 281

- BoxOverlapQuery, 283
- BoxOverlapQueryParams, 285
- Failed
 - Fusion, 49
- FailedClientCantSpawn
 - Fusion, 52
- FailedLocalPlayerNotYetSet
 - Fusion, 52
- FailedToCreateInstance
 - Fusion, 52
- FailedToLoadPrefabSynchronously
 - Fusion, 52
- FieldEditorButtonAttribute, 158
 - AllowMultipleTargets, 159
 - FieldEditorButtonAttribute, 159
 - Label, 160
 - TargetMethod, 160
- FieldsMask
 - FieldsMask< T >, 161, 162
- FieldsMask< T >, 160
 - FieldsMask, 161, 162
 - Mask, 163
 - operator Mask256, 162
- Find
 - NetConnectionMap, 1033, 1034
- FindByIndex
 - NetConnectionMap, 1034
- FindObject
 - NetworkRunner, 645
- FindObjectOfTypeInOrder< T >
 - NestedComponentUtilities, 366
- FindObjectOfTypeInOrder< T, TCast >
 - NestedComponentUtilities, 368
- FirstChild
 - Fusion, 61
- FixedArray
 - FixedArray< T >, 164
- FixedArray< T >, 163
 - Clear, 165
 - CopyFrom, 165
 - CopyTo, 166
 - Create< T >, 166
 - Create< TActual, TAdapted >, 167
 - CreateFromFieldSequence< T >, 167
 - FixedArray, 164
 - GetEnumerator, 168
 - IndexOf< T >, 168
 - Length, 169
 - this[int index], 169
 - ToArray, 169
 - ToListString, 169
 - ToString, 169
- FixedArray< T >.Enumerator, 170
 - Current, 171
 - Dispose, 171
 - Enumerator, 170
 - MoveNext, 171
 - Reset, 171
- FixedBufferPropertyAttribute, 172
 - Capacity, 173
 - FixedBufferPropertyAttribute, 172
 - SurrogateType, 173
 - Type, 173
- FixedStorage, 173
 - GetWordCount< T >, 173
- FixedUpdateNetwork
 - NetworkBehaviour, 388
 - SimulationBehaviour, 926
- FixedUpdateNetworkExecutionCount
 - BehaviourStatisticsSnapshot, 1078
- FixedUpdateNetworkExecutionTime
 - BehaviourStatisticsSnapshot, 1078
- FLAG_ADDRESSABLE
 - SceneRef, 885
- FLAG_DUMMY
 - SimulationMessage, 949
- FLAG_INTERNAL
 - SimulationMessage, 949
- FLAG_NOT_TICK_ALIGNED
 - SimulationMessage, 950
- FLAG_REMOTE
 - SimulationMessage, 950
- FLAG_STATIC
 - SimulationMessage, 950
- FLAG_TARGET_PLAYER
 - SimulationMessage, 950
- FLAG_TARGET_SERVER
 - SimulationMessage, 950
- FLAG_UNRELIABLE
 - SimulationMessage, 950
- FLAG_USER_FLAGS_START
 - SimulationMessage, 951
- FLAG_USER_MESSAGE
 - SimulationMessage, 951
- Flags
 - NetworkObject, 534
 - NetworkObjectHeader, 555
 - NetworkObjectMeta, 560
 - SimulationMessage, 951
- FLAGS_RESERVED
 - SimulationMessage, 951
- FLAGS_RESERVED_BITS
 - SimulationMessage, 951
- Float
 - NetworkBehaviourBufferInterpolator, 426
- FloatCompressed, 174
 - Equals, 175
 - GetHashCode, 175
 - operator float, 175
 - operator FloatCompressed, 176
 - operator!=, 176
 - operator==, 177
 - valueEncoded, 177
- FloatUtils, 177
 - Compress, 178
 - Decompress, 178

- DEFAULT_ACCURACY, [179](#)
- FloorToInt
 - Maths, [326](#)
- ForceRemoteRenderTimeframe
 - NetworkObject, [534](#)
- Forward
 - Fusion, [60](#)
- ForwardTicks
 - FusionStatisticsSnapshot, [1081](#)
- Frames
 - Fusion, [60](#)
- FramesPerSecond
 - Fusion, [60](#)
- Free
 - DynamicHeap, [133](#)
 - DynamicHeapInstance, [139](#)
 - Native, [345](#)
 - NetConnectionMap, [1032](#)
- From
 - Fusion, [53](#)
 - NetworkBehaviourBufferInterpolator, [433](#)
- FromActorId
 - NetAddress, [971](#)
- FromAsyncOperation
 - NetworkSceneAsyncOp, [706](#)
- FromCompleted
 - NetworkSceneAsyncOp, [707](#)
- FromCoroutine
 - NetworkSceneAsyncOp, [707](#)
- FromCustom
 - NetworkObjectTypeId, [574](#)
- FromEncoded
 - PlayerRef, [773](#)
- FromError
 - NetworkSceneAsyncOp, [708](#)
- FromIndex
 - NetworkPrefabId, [587](#)
 - PlayerRef, [773](#)
 - SceneRef, [882](#)
- FromInts
 - ReliableKey, [1063](#)
- FromLocal
 - RpclInfo, [868](#)
- FromMessage
 - RpclInfo, [868](#)
- FromPath
 - SceneRef, [882](#)
- FromPrefabId
 - NetworkObjectTypeId, [574](#)
- FromRaw
 - NetworkPrefabId, [588](#)
 - SceneRef, [883](#)
- FromSceneObjectId
 - NetworkObjectTypeId, [575](#)
- FromSceneRefAndObjectIndex
 - NetworkObjectTypeId, [575](#)
- FromStruct
 - NetworkObjectTypeId, [576](#)
- FromTask
 - NetworkSceneAsyncOp, [708](#)
- FromULongs
 - ReliableKey, [1063](#)
- Full
 - NetConnectionMap, [1036](#)
 - SimulationConfig, [931](#)
 - SimulationInput.Buffer, [939](#)
- FullCone
 - Fusion.Sockets.Stun, [70](#)
- Fusion, [33](#)
 - After, [61](#)
 - All, [57](#)
 - AllowStateAuthorityOverride, [50](#)
 - AlreadyRunning, [59](#)
 - Always, [48](#)
 - AreaOfInterest, [50](#)
 - AuthenticationTicketExpired, [59](#)
 - AutoHostOrClient, [48](#)
 - Before, [61](#)
 - BitwiseAndNotEqualZero, [47](#)
 - Box, [49](#)
 - Bytes, [60](#)
 - Capsule, [49](#)
 - Client, [48](#), [59](#)
 - ClientServer, [58](#), [60](#)
 - CompareOperator, [47](#)
 - ConnectionRefused, [59](#)
 - ConnectionTimeout, [59](#)
 - ConnectionType, [47](#)
 - ConterMask, [51](#)
 - Count, [60](#)
 - Custom, [52](#), [58](#)
 - CustomAuthenticationFailed, [59](#)
 - Degrees, [60](#)
 - DegreesPerSecond, [60](#)
 - DestroyWhenStateAuthorityLeaves, [50](#)
 - Direct, [47](#)
 - DisconnectedByPluginLogic, [59](#)
 - DontDestroyOnLoad, [50](#), [51](#)
 - DrawIfMode, [47](#)
 - EditMode, [48](#)
 - EditorButtonVisibility, [48](#)
 - Equal, [47](#)
 - Error, [58](#)
 - Failed, [49](#)
 - FailedClientCantSpawn, [52](#)
 - FailedLocalPlayerNotYetSet, [52](#)
 - FailedToCreateInstance, [52](#)
 - FailedToLoadPrefabSynchronously, [52](#)
 - FirstChild, [61](#)
 - Forward, [60](#)
 - Frames, [60](#)
 - FramesPerSecond, [60](#)
 - From, [53](#)
 - FusionGlobalScriptableObjectUnloadDelegate, [61](#)
 - GameClosed, [59](#)
 - GameIdAlreadyExists, [59](#)

GameIsFull, 59
GameMode, 48
GameNotFound, 59
GlobalObjectInterest, 50
Greater, 47
GreaterOrEqual, 47
HasMainNetworkTRSP, 50
Hide, 48
High, 53
HitboxTypes, 48
HitOptions, 49
Host, 48, 59
HostMigration, 59
Ignore, 49, 50
IgnoreInputAuthority, 49
IncludeBox2D, 49
IncludePhysX, 49
IncompatibleConfiguration, 58
Initial, 51
InProgress, 50
InputAuthority, 57
InsufficientSourceAuthority, 54, 56
InsufficientTargetAuthority, 54
InternalStruct, 52
Interpolated, 53
Invalid, 52, 58
InvalidArguments, 59
InvalidAuthentication, 59
InvalidRegion, 59
Invoked, 54
IsZero, 47
Kilobytes, 60
LastChild, 61
Latest, 53
Less, 47
LessOrEqual, 47
LoadError, 50
Local, 53
Low, 53
Lowest, 53
MaskBroadcast, 56
MaskCulled, 56
MaskNotSent, 56
MaskSent, 56
MaskVersion, 50
MasterClientObject, 50
MaxCcuReached, 59
Medium, 53
Megabytes, 60
MilliSecs, 60
Multiplier, 60
NetworkObjectAcquireResult, 49
NetworkObjectFlags, 49
NetworkObjectHeaderFlags, 50
NetworkObjectSpawnDelegate, 61
NetworkPrefabTableGetPrefabResult, 50
NetworkSceneInfoChangeSource, 51
NetworkSceneInfoDefaultFlags, 51
NetworkSpawnFlags, 51
NetworkSpawnStatus, 51
NetworkTypeIdKind, 52
NoActiveConnections, 56
None, 47, 49–51, 56, 60
Normalized, 60
NormalizedPercentage, 60
NotCulled, 56
NotEqual, 47
NotFound, 50
NotInvokableDuringResim, 54, 56
NotInvokableLocally, 54
NotSentBroadcastNoActiveConnections, 56
NotSentBroadcastNoConfirmedNorInterested-
Clients, 56
NotSentTargetClientNotAvailable, 56
NotSentTargetObjectNotConfirmed, 56
NotSentTargetObjectNotInPlayerInterest, 56
NotZero, 47
Ok, 58
OperationCanceled, 59
OperationTimeout, 59
Packets, 60
PageSizes, 52
PayloadSizeExceeded, 54, 56
Percentage, 60
PerSecond, 60
Photon, 58
PhotonCloudTimeout, 59
Player, 53
PlayMode, 48
Prefab, 52
PriorityLevel, 52
Proxies, 57
Queued, 52
Radians, 60
RadiansPerSecond, 60
ReadOnly, 48
Relayed, 47
Reliable, 54
Remote, 51, 53, 57
RenderSource, 53
RenderTimeframe, 53
Resimulate, 60
Retry, 49
RpcChannel, 53
RpcHostMode, 54
RpcInvokeDelegate, 61
RpcLocalInvokeResult, 54
RpcSendCullResult, 54
RpcSendMessageResult, 56
RpcSources, 56
RpcStaticInvokeDelegate, 62
RpcTargets, 57
RpcTargetStatus, 57
SceneCountMask, 51
SceneObject, 52
ScriptHeaderBackColor, 57

- ScriptHeaderIcon, [57](#)
- ScriptHeaderStyle, [58](#)
- Seconds, [60](#)
- Self, [57](#)
- SentBroadcast, [56](#)
- SentToServerForForwarding, [56](#)
- SentToTargetClient, [56](#)
- Server, [48](#), [59](#)
- ServerInRoom, [59](#)
- SessionLobby, [58](#)
- Shared, [48](#), [58](#), [60](#)
- SharedModeStateAuthLocalPlayer, [51](#)
- SharedModeStateAuthMasterClient, [51](#)
- ShutdownReason, [58](#)
- SimulationModes, [59](#)
- SimulationStages, [59](#)
- Single, [48](#)
- SourcelsHostPlayer, [54](#)
- SourcelsServer, [54](#)
- Spawned, [52](#)
- SpawnedByClient, [50](#)
- Sphere, [49](#)
- SquareMagnitude, [60](#)
- StateAuthority, [57](#)
- Struct, [50](#)
- StructArray, [50](#)
- SubtickAccuracy, [49](#)
- Success, [49](#), [50](#)
- TargetPlayerIsNotLocal, [54](#)
- TargetPlayerIsLocalButRpcIsNotInvokableLocally, [56](#)
- TargetPlayerIsNotLocal, [54](#)
- TargetPlayerUnreachable, [56](#)
- Ticks, [60](#)
- TicksPerSecond, [60](#)
- To, [53](#)
- Topologies, [60](#)
- Units, [60](#)
- Unity, [58](#)
- UnityPlayerLoopSystemAddMode, [61](#)
- Unreachable, [57](#)
- Unreliable, [54](#)
- V1, [50](#)
- Fusion.Analyzer, [62](#)
 - Manual, [63](#)
 - None, [63](#)
 - ResetMethod, [63](#)
 - StaticFieldResetMode, [62](#)
- Fusion.Async, [63](#)
- Fusion.Encryption, [63](#)
- Fusion.Internal, [63](#)
- Fusion.LagCompensation, [64](#)
 - Box2D, [65](#)
 - Hitbox, [65](#)
 - HitType, [65](#)
 - None, [65](#)
 - PhysX, [65](#)
 - PreProcessingDelegate, [65](#)
- Fusion.Protocol, [66](#)
 - DisconnectReason, [66](#)
 - Error, [66](#)
 - IncompatibleConfiguration, [66](#)
 - InvalidEventCode, [66](#)
 - InvalidJoinGameMode, [66](#)
 - InvalidJoinMsgType, [66](#)
 - None, [66](#)
 - ServerAlreadyInRoom, [66](#)
 - ServerLogic, [66](#)
- Fusion.Runtime, [67](#)
- Fusion.Runtime.Unity, [67](#)
- Fusion.Sockets, [67](#)
 - ByRemote, [69](#)
 - Connected, [69](#)
 - Connecting, [69](#)
 - Created, [69](#)
 - Disconnected, [69](#)
 - NetCommands, [68](#)
 - NetConnectFailedReason, [68](#)
 - NetConnectionStatus, [69](#)
 - NetDisconnectReason, [69](#)
 - NetPacketType, [69](#)
 - Ok, [70](#)
 - OnConnectionRequestReply, [69](#)
 - Refuse, [70](#)
 - Requested, [69](#)
 - SendWindowFull, [69](#)
 - SequenceOutOfBounds, [69](#)
 - ServerFull, [69](#)
 - ServerRefused, [69](#)
 - Shutdown, [69](#)
 - Timeout, [69](#)
 - Unknown, [69](#)
 - Waiting, [70](#)
- Fusion.Sockets.Stun, [70](#)
 - FullCone, [70](#)
 - Invalid, [70](#)
 - NATType, [70](#)
 - OpenInternet, [70](#)
 - Symmetric, [70](#)
 - UdpBlocked, [70](#)
- Fusion.Statistics, [71](#)
- FusionGlobalScriptableObject< T >, [179](#)
 - GlobalInternal, [181](#)
 - IsGlobal, [181](#)
 - IsGlobalLoadedInternal, [181](#)
 - OnDisable, [180](#)
 - OnLoadedAsGlobal, [180](#)
 - OnUnloadedAsGlobal, [180](#)
 - TryGetGlobalInternal, [180](#)
 - UnloadGlobalInternal, [181](#)
- FusionGlobalScriptableObjectAttribute, [182](#)
 - DefaultContents, [183](#)
 - DefaultContentsGeneratorMethod, [183](#)
 - DefaultPath, [183](#)
 - FusionGlobalScriptableObjectAttribute, [182](#)
- FusionGlobalScriptableObjectLoadResult, [183](#)

- FusionGlobalScriptableObjectLoadResult, 184
 - Object, 184
 - operator FusionGlobalScriptableObjectLoadResult, 184
 - Unloader, 184
- FusionGlobalScriptableObjectSourceAttribute, 185
 - AllowEditMode, 186
 - AllowFallback, 186
 - Load, 186
 - ObjectType, 186
 - Order, 186
- FusionGlobalScriptableObjectUnloadDelegate
 - Fusion, 61
- FusionMonoBehaviour, 187
- FusionScriptableObject, 187
- FusionStatisticsManager, 1079
 - CompleteSnapshot, 1079
 - ObjectStatisticsManager, 1079
- FusionStatisticsSnapshot, 1079
 - ForwardTicks, 1081
 - GeneralAllocMemoryFreeInBytes, 1081
 - GeneralAllocMemoryUsedInBytes, 1081
 - InBandwidth, 1081
 - InObjectUpdates, 1081
 - InPackets, 1081
 - InputInBandwidth, 1082
 - InputOutBandwidth, 1082
 - InputReceiveDelta, 1082
 - InterpolationOffset, 1082
 - InterpolationSpeed, 1082
 - ObjectsAllocMemoryFreeInBytes, 1082
 - ObjectsAllocMemoryUsedInBytes, 1083
 - OutBandwidth, 1083
 - OutObjectUpdates, 1083
 - OutPackets, 1083
 - Resimulations, 1083
 - RoundTripTime, 1083
 - SimulationSpeed, 1084
 - SimulationTimeOffset, 1084
 - StateReceiveDelta, 1084
 - TimeResets, 1084
 - WordsReadCount, 1084
 - WordsReadSize, 1084
 - WordsWrittenCount, 1085
 - WordsWrittenSize, 1085
- GameClosed
 - Fusion, 59
- GameAlreadyExists
 - Fusion, 59
- GameIsFull
 - Fusion, 59
- GameMode
 - Fusion, 48
 - HostMigrationToken, 214
 - NetworkRunner, 690
 - StartGameArgs, 1072
- GameNotFound
 - Fusion, 59
- GameObject
 - LagCompensatedHit, 278
- GeneralAllocMemoryFreeInBytes
 - FusionStatisticsSnapshot, 1081
- GeneralAllocMemoryUsedInBytes
 - FusionStatisticsSnapshot, 1081
- GenerateKey
 - IDataEncryption, 145
- Get
 - NetworkArray< T >, 377
 - NetworkDictionary< K, V >, 477
 - NetworkDictionaryReadOnly< K, V >, 483
 - NetworkLinkedList< T >, 511
 - NetworkLinkedListReadOnly< T >, 518
 - NetworkString< TSize >, 743
 - SimulationInput.Buffer, 938
 - TickRate, 1107
- Get< T >
 - NetworkInput, 502
- GetAlignment
 - Native, 345
- GetAlignment< T >
 - Native, 346
- GetAllBehaviours
 - NetworkRunner, 646
- GetAllBehaviours< T >
 - NetworkRunner, 646
- GetAllNetworkBehaviourTypes
 - ReflectionUtils, 854
- GetAllNetworkObjects
 - NetworkRunner, 647
- GetAllSimulationBehaviourTypes
 - ReflectionUtils, 855
- GetAllWeavedAssemblies
 - ReflectionUtils, 855
- GetAllWeavedNetworkBehaviourTypes
 - ReflectionUtils, 855
- GetAllWeavedSimulationBehaviourTypes
 - ReflectionUtils, 855
- GetAllWeaverGeneratedTypes
 - ReflectionUtils, 856
- GetAreaOfInterestGizmoData
 - NetworkRunner, 647
 - Simulation, 908
- GetArrayReader< T >
 - NetworkBehaviour, 388
- GetAvailableRegions
 - NetworkRunner, 648
- GetAwaiter
 - NetworkSceneAsyncOp, 708
 - NetworkSpawnOp, 729
- GetBehaviour< T >
 - Behaviour, 111
- GetBehaviourChangedTickArray
 - NetworkObjectHeader, 552
- GetBehaviourReader< T >
 - NetworkBehaviour, 389, 390
- GetBehaviourReader< TBehaviour, TProperty >

- NetworkBehaviour, [390](#)
- GetBit
 - Mask256, [314](#)
- GetByteArrayHashCode
 - ReadWriteUtilsForWeaver, [847](#)
- GetByteCountUtf8NoHash
 - ReadWriteUtilsForWeaver, [847](#)
- GetCapacity< TSize >
 - NetworkString< TSize >, [744](#)
- GetCellSize
 - Simulation.AreaOfInterest, [922](#)
- GetChangeDetector
 - NetworkBehaviour, [391](#)
- GetCharCount
 - NetworkString< TSize >, [744](#)
- GetConfigPointer
 - NetPeer, [1041](#)
- GetConnection
 - NetPeerGroup, [1051](#)
- GetConnectionByIndex
 - NetPeerGroup, [1051](#)
- GetConnectionIdleTime
 - NetPeerGroup, [1051](#)
- GetCurrentVersion
 - Versioning, [1157](#)
- GetCurveValue
 - NetConfigSimulationOscillator, [1024](#)
- GetCustomAttributeOrThrow< T >
 - ReflectionUtils, [856](#)
- GetData
 - ReliableHeader, [1058](#)
 - SimulationMessage, [945](#)
- GetDataPointer
 - NetBitBuffer, [980](#)
 - NetworkObjectHeader, [552](#)
- GetDataWordCount
 - NetworkObjectHeader, [553](#)
- GetDictionaryReader< K, V >
 - NetworkBehaviour, [391](#), [392](#)
- GetDivisor
 - TickRate, [1107](#)
- GetElementHashCode
 - IElementReaderWriter< T >, [228](#)
- GetElementWordCount
 - IElementReaderWriter< T >, [229](#)
- GetEntries
 - NetworkPrefabTable, [604](#)
- GetEnumerator
 - BVHDraw, [286](#)
 - FixedArray< T >, [168](#)
 - HitboxColliderContainerDraw, [290](#)
 - NetworkArray< T >, [377](#)
 - NetworkBehaviour.ChangeDetector.Enumerable, [410](#)
 - NetworkDictionary< K, V >, [477](#)
 - NetworkLinkedList< T >, [511](#)
 - NetworkString< TSize >, [744](#)
 - SerializableDictionary< TKey, TValue >, [891](#)
 - SnapshotHistoryDraw, [305](#)
- GetExecutionOrder
 - NetworkProjectConfig, [614](#)
- GetFieldOffset
 - Native, [346](#)
- GetFlag
 - SimulationMessage, [945](#)
- GetGroup
 - NetPeer, [1041](#)
- GetGuid
 - NetworkPrefabTable, [604](#)
- GetHashCode
 - Allocator.Config, [85](#)
 - Angle, [93](#)
 - FloatCompressed, [175](#)
 - Mask256, [314](#)
 - NetAddress, [971](#)
 - NetAddress.EqualityComparer, [974](#)
 - NetConnectionId, [1029](#)
 - NetConnectionId.EqualityComparer, [1030](#)
 - NetworkBehaviourId, [435](#)
 - NetworkBool, [458](#)
 - NetworkButtons, [461](#)
 - NetworkId, [497](#)
 - NetworkId.EqualityComparer, [501](#)
 - NetworkLoadSceneParameters, [521](#)
 - NetworkObjectGuid, [545](#)
 - NetworkObjectGuid.EqualityComparer, [550](#)
 - NetworkObjectHeader, [553](#)
 - NetworkObjectNestingKey, [563](#)
 - NetworkObjectNestingKey.EqualityComparer, [566](#)
 - NetworkObjectTypeId, [576](#)
 - NetworkObjectTypeId.EqualityComparer, [581](#)
 - NetworkPrefabId, [588](#)
 - NetworkPrefabId.EqualityComparer, [591](#)
 - NetworkPrefabRef, [596](#)
 - NetworkPrefabRef.EqualityComparer, [601](#)
 - NetworkRunnerUpdaterDefaultInvokeSettings, [703](#)
 - NetworkSceneInfo, [714](#)
 - NetworkSceneLoadId, [718](#)
 - NetworkSceneObjectId, [722](#)
 - NetworkString< TSize >, [744](#)
 - PlayerRef, [775](#)
 - Ptr, [830](#)
 - Ptr.EqualityComparer, [833](#)
 - QuaternionCompressed, [836](#)
 - SceneRef, [883](#)
 - SerializableType< BaseType >, [897](#)
 - Tick, [1094](#)
 - Tick.EqualityComparer, [1099](#)
 - Vector2Compressed, [1142](#)
 - Vector3Compressed, [1147](#)
 - Vector4Compressed, [1152](#)
- GetId
 - NetworkPrefabTable, [604](#)
- GetInput< T >
 - NetworkBehaviour, [392](#)
- GetInputAuthority

- Simulation, 909
- GetInputForPlayer
 - Simulation, 909
- GetInputForPlayer< T >
 - NetworkRunner, 648
- GetInsertTime
 - SimulationInput.Buffer, 938
- GetInstancesCount
 - NetworkPrefabTable, 605
- GetInstancesEnumerator
 - NetworkRunner, 648
- GetInterfaceListHead
 - NetworkRunner, 648
- GetInterfaceListNext
 - NetworkRunner, 650
- GetInterfaceListPrev
 - NetworkRunner, 650
- GetInterfaceListsCount
 - NetworkRunner, 650
- GetInts
 - ReliableKey, 1064
- GetLastUsedInputHeader
 - SimulationInput.Buffer, 939
- GetLength
 - NetBitBuffer.Offset, 999
 - UTF32Tools, 1137
- GetLengthPrefixedUTF8ByteCount
 - Native, 346
- GetLinkListReader< T >
 - NetworkBehaviour, 393
- GetLocalAuthorityMask
 - NetworkBehaviour, 394
 - NetworkObject, 530
- GetMainNetworkTRSPData
 - NetworkObjectHeader, 553
- GetMaxAlignment
 - Native, 347, 348
- GetMaxWordCount
 - NetworkInputUtils, 505
- GetMemorySnapshot
 - NetworkRunner, 651
- GetMetaData
 - NetworkBehaviourUtils, 442
- GetNestedComponentInChildren< T, TStopOn >
 - NestedComponentUtilities, 369
- GetNestedComponentInParent< T, TStopOn >
 - NestedComponentUtilities, 370
- GetNestedComponentInParents< T, TStopOn >
 - NestedComponentUtilities, 370
- GetNestedComponentsInChildren< T >
 - NestedComponentUtilities, 370
- GetNestedComponentsInChildren< T, TSearch, TStop >
 - NestedComponentUtilities, 371
- GetNestedComponentsInChildren< T, TStopOn >
 - NestedComponentUtilities, 371
- GetNestedComponentsInParents< T >
 - NestedComponentUtilities, 371
- GetNestedComponentsInParents< T, TStop >
 - NestedComponentUtilities, 372
- GetNetworkObjectStatistics
 - NetworkObjectStatisticsManager, 1090
- GetNextPrime
 - Primes, 780, 781
- GetNotifyDataBuffer
 - NetPeerGroup, 1052
- GetObjectsAndPlayersInAreaOfInterestCell
 - Simulation, 909
- GetObjectsInAreaOfInterestForPlayer
 - NetworkRunner, 651
 - Simulation, 909
- GetOffset
 - NetBitBuffer, 980
- GetParentComponent< T >
 - NestedComponentUtilities, 372
- GetPhysicsScene
 - NetworkRunner, 651
- GetPhysicsScene2D
 - NetworkRunner, 652
- GetPlayerActorId
 - NetworkRunner, 652
- GetPlayerConnectionToken
 - NetworkRunner, 652
- GetPlayerConnectionType
 - NetworkRunner, 653
- GetPlayerObject
 - NetworkRunner, 653
- GetPlayerRtt
 - NetworkRunner, 653
- GetPlayerTickAndAlpha
 - HitboxManager, 196
- GetPlayerUserId
 - NetworkRunner, 654
- GetPrefabId
 - INetworkObjectProvider, 239
- GetPressed
 - NetworkButtons, 462
- GetPropertyReader< T >
 - NetworkBehaviour, 394
- GetPropertyReader< TBehaviour, TProperty >
 - NetworkBehaviour, 395
- GetRawInputForPlayer
 - NetworkRunner, 654
- GetRef< T >
 - NetworkArrayExtensions, 381
- GetReleased
 - NetworkButtons, 462
- GetRenderBuffers
 - RenderTimeline, 859
- GetResult
 - NetworkSceneAsyncOp.Awaiter, 711
 - NetworkSpawnOp.Awaiter, 731
- GetResumeSnapshotNetworkObjects
 - NetworkRunner, 654
- GetResumeSnapshotNetworkSceneObjects
 - NetworkRunner, 654

- GetRpcStaticIndexOrThrow
 - NetworkBehaviourUtils, 442
- GetRpcTargetStatus
 - NetworkRunner, 655
- GetRunnerForGameObject
 - NetworkRunner, 655
- GetRunnerForScene
 - NetworkRunner, 655
- GetSceneRef
 - INetworkSceneManager, 250
- GetShortAssemblyQualifiedName
 - SerializableType< BaseType >, 897
- GetSingleton< T >
 - NetworkRunner, 656
- GetSource
 - NetworkPrefabTable, 605, 606
- GetStateAuthority
 - Simulation, 910
- GetStaticWordCount
 - NetworkBehaviourUtils, 443
- GetStatisticsSnapshot
 - HitboxManager, 196
- GetStringHashCode
 - ReadWriteUtilsForWeaver, 848
- GetTickRate
 - TickRate, 1108
- GetType
 - NetworkInputUtils, 505
- GetTypeKey
 - NetworkInputUtils, 506
- GetUlongs
 - ReliableKey, 1064
- GetUnreliableDataBuffer
 - NetPeerGroup, 1052
- GetVersion
 - NetworkObjectFlagsExtensions, 539
- GetWeavedAttributeOrThrow
 - ReflectionUtils, 857
- GetWordCount
 - NetworkBehaviourUtils, 443
 - NetworkInputUtils, 506
 - NetworkObject, 530
- GetWordCount< T >
 - FixedStorage, 173
 - NetworkStructUtils, 758
- GetWordCountString
 - ReadWriteUtilsForWeaver, 848
- GizmosColor
 - Hitbox, 192
 - HitboxRoot, 211
- GizmosDrawWireCapsule
 - LagCompensationDraw, 291
- Global
 - NetworkProjectConfig, 620
 - NetworkProjectConfigAsset, 622
- GlobalInternal
 - FusionGlobalScriptableObject< T >, 181
- GlobalObjectInterest
 - Fusion, 50
- GlobalsSize
 - Allocator.Config, 86
 - HeapConfiguration, 188
- Greater
 - Fusion, 47
- GreaterOrEqual
 - Fusion, 47
- Group
 - NetConnectionId, 1030
 - NetPeerGroup, 1055
- GroupCount
 - NetPeer, 1046
- GroupIndex
 - NetConnectionId, 1030
- GroupTypesByNamespace
 - SerializeReferenceTypePickerAttribute, 901
- Guid
 - NetworkObjectPrefabData, 566
- Handle
 - NetSocket, 1057
- HasAddress
 - NetAddress, 972
- HasAnyActiveConnections
 - Simulation, 910
- HasHeader
 - NetworkPrefabAcquireContext, 585
 - NetworkPrefabInfo, 592
- HasInputAuthority
 - NetworkBehaviour, 404
 - NetworkObject, 535
- HasMainNetworkTRSP
 - Fusion, 50
 - NetworkObjectHeader, 554
- HasSingleton< T >
 - NetworkRunner, 656
- HasStateAuthority
 - NetworkBehaviour, 404
 - NetworkObject, 535
- HasStaticWordCount
 - NetworkBehaviourUtils, 443
- Head
 - ReliableList, 1067
- Header
 - NetworkPrefabInfo, 592
 - SimulationInput, 935
- Heap
 - NetworkProjectConfig, 617
- HEAP_ALIGNMENT
 - Allocator, 82
- HeapConfiguration, 187
 - GlobalsSize, 188
 - Init, 188
 - PageCount, 188
 - PageShift, 188
 - Tostring, 188
- HeapSizeAllocated
 - Allocator.Config, 87

- HeapSizeUsable
 - Allocator.Config, 87
- HexToBytes
 - BinUtils, 114
- Hide
 - DrawIfAttribute, 131
 - Fusion, 48
 - ScriptHelpAttribute, 887
- HideArrayElementLabelAttribute, 189
 - HideArrayElementLabelAttribute, 189
- HideNetworkObjectInactivityGuard
 - NetworkProjectConfig, 617
- High
 - Fusion, 53
- Hitbox, 189
 - BoxExtents, 191
 - CapsuleExtents, 191
 - CapsuleRadius, 192
 - ColliderIndex, 193
 - DrawGizmos, 191
 - Fusion.LagCompensation, 65
 - GizmosColor, 192
 - HitboxActive, 193
 - HitboxIndex, 193
 - LagCompensatedHit, 278
 - Offset, 192
 - OnDrawGizmos, 191
 - Position, 193
 - PositionRotationQueryParams, 294
 - Root, 192
 - SetLayer, 191
 - SphereRadius, 192
 - Type, 192
- HitboxActive
 - Hitbox, 193
- HitboxBufferLengthInMs
 - LagCompensationSettings, 310
- HitboxColliderContainerDraw, 289
 - GetEnumerator, 290
- HitboxColliderPosition
 - LagCompensatedHit, 278
- HitboxColliderRotation
 - LagCompensatedHit, 278
- HitboxDefaultCapacity
 - LagCompensationSettings, 310
- Hitboxes
 - HitboxRoot, 212
- HitboxesCount
 - LagCompensationStatisticsSnapshot, 1086
- HitboxIndex
 - Hitbox, 193
- HitboxManager, 193
 - BVHDepth, 206
 - BVHNodes, 207
 - DrawInfo, 207
 - GetPlayerTickAndAlpha, 196
 - GetStatisticsSnapshot, 196
 - OverlapBox, 196–198
 - OverlapSphere, 198–200
 - PositionRotation, 201
 - Raycast, 202, 203
 - RaycastAll, 204–206
 - TotalHitboxes, 207
- HitboxRoot, 207
 - BroadRadius, 211
 - Config, 211
 - ConfigFlags, 209
 - Default, 209
 - DrawGizmos, 209
 - GizmosColor, 211
 - Hitboxes, 212
 - HitboxRootActive, 212
 - IncludeInactiveHitboxes, 209
 - InInterest, 212
 - InitHitboxes, 209
 - IsHitboxActive, 210
 - Legacy, 209
 - Manager, 213
 - MAX_HITBOXES, 212
 - Offset, 212
 - OnDrawGizmos, 210
 - ReinitializeHitboxesBeforeRegistration, 209
 - SetHitboxActive, 210
 - SetMinBoundingRadius, 211
- HitboxRootActive
 - HitboxRoot, 212
- HitboxTypes
 - Fusion, 48
- HitOptions
 - Fusion, 49
- HitType
 - Fusion.LagCompensation, 65
- Host
 - Fusion, 48, 59
- HostMigration
 - Fusion, 59
 - NetworkProjectConfig, 617
 - SimulationConfig, 931
- HostMigrationConfig, 213
 - EnableAutoUpdate, 213
 - UpdateDelay, 213
- HostMigrationResume
 - StartGameArgs, 1072
- HostMigrationToken, 214
 - GameMode, 214
 - StartGameArgs, 1072
- HostMode
 - RpcAttribute, 862
- HostPlayer
 - SimulationRuntimeConfig, 954
- IAfterAllTicks, 214
 - AfterAllTicks, 215
- IAfterClientPredictionReset, 216
 - AfterClientPredictionReset, 216
- IAfterHostMigration, 216
 - AfterHostMigration, 217

- IAfterRender, 217
 - AfterRender, 217
- IAfterSpawned, 218
 - AfterSpawned, 218
- IAfterTick, 218
 - AfterTick, 219
- IAfterUpdate, 219
 - AfterUpdate, 219
- IAfterUpdateRemotePrefabs, 219
 - AfterUpdateRemotePrefabs, 220
- IAsyncOperation, 220
 - Completed, 221
 - Error, 221
 - IsDone, 221
- IBeforeAllTicks, 221
 - BeforeAllTicks, 222
- IBeforeClientPredictionReset, 222
 - BeforeClientPredictionReset, 222
- IBeforeCopyPreviousState, 223
 - BeforeCopyPreviousState, 223
- IBeforeHitboxRegistration, 223
 - BeforeHitboxRegistration, 224
- IBeforeSimulation, 224
 - BeforeSimulation, 224
- IBeforeTick, 225
 - BeforeTick, 225
- IBeforeUpdate, 225
 - BeforeUpdate, 226
- IBeforeUpdateRemotePrefabs, 226
 - BeforeUpdateRemotePrefabs, 226
- ICommunicator, 824
 - CommunicatorID, 827
 - Poll, 825
 - PushPackage, 825
 - ReceivePackage, 825
 - RegisterPackageCallback< T >, 826
 - SendMessage, 826
 - SendPackage, 827
 - Service, 827
- ICoroutine, 227
- Id
 - NetworkBehaviour, 404
 - NetworkObject, 536
 - NetworkObjectHeader, 555
 - NetworkObjectHeaderPtr, 558
 - NetworkObjectMeta, 560
 - ReliableHeader, 1059
- IDataEncryption, 143
 - ComputeHash, 143
 - DecryptData, 144
 - EncryptData, 144
 - GenerateKey, 145
 - Setup, 145
 - VerifyHash, 145
- IDespawned, 227
 - Despawned, 227
- IElementReaderWriter< T >, 228
 - GetElementHashCode, 228
 - GetElementWordCount, 229
 - Read, 229
 - ReadRef, 229
 - Write, 230
- IExportedWordCount, 1159
 - WordCount, 1159
- IFixedStorage, 230
- Ignore
 - Fusion, 49, 50
- IgnoreInputAuthority
 - Fusion, 49
- IInputAuthorityGained, 230
 - InputAuthorityGained, 231
- IInputAuthorityLost, 231
 - InputAuthorityLost, 231
- IInterestEnter, 231
 - InterestEnter, 232
- IInterestExit, 232
 - InterestExit, 232
- ILocalPrefabCreated, 233
 - LocalPrefabCreated, 233
- IMessage, 827
- InBandwidth
 - FusionStatisticsSnapshot, 1081
 - NetworkObjectStatisticsSnapshot, 1091
- IncludeBox2D
 - Fusion, 49
- IncludeInactiveHitboxes
 - HitboxRoot, 209
- IncludePhysX
 - Fusion, 49
- IncompatibleConfiguration
 - Fusion, 58
 - Fusion.Protocol, 66
- IncrementPost
 - AtomicInt, 108
- IncrementPre
 - AtomicInt, 108
- IndexOf
 - NetworkLinkedList< T >, 511
 - NetworkLinkedListReadOnly< T >, 518
 - NetworkSceneInfo, 714
 - NetworkString< TSize >, 745–747
- IndexOf< T >
 - FixedArray< T >, 168
 - NetworkArrayExtensions, 381
- IndexOf< TOtherSize >
 - NetworkString< TSize >, 748–750
- InEditMode
 - ReadOnlyAttribute, 841
- INetBitWriteStream, 955
 - OffsetBits, 958
 - WriteBoolean, 956
 - WriteBytesAligned, 956
 - WriteInt32, 956
 - WriteInt32VarLength, 957
 - WriteUInt64VarLength, 957
- INetPeerGroupCallbacks, 958

- OnConnected, [959](#)
- OnConnectionAttempt, [959](#)
- OnConnectionFailed, [959](#)
- OnConnectionRequest, [960](#)
- OnDisconnected, [960](#)
- OnNotifyData, [960](#)
- OnNotifyDelivered, [962](#)
- OnNotifyDispose, [962](#)
- OnNotifyLost, [962](#)
- OnReliableData, [963](#)
- OnUnconnectedData, [963](#)
- OnUnreliableData, [963](#)
- INetSocket, [964](#)
 - Bind, [964](#)
 - CanFragment, [965](#)
 - Create, [965](#)
 - DeleteEncryptionKey, [965](#)
 - Destroy, [966](#)
 - Initialize, [966](#)
 - Poll, [966](#)
 - Receive, [967](#)
 - Send, [967](#)
 - SetupEncryption, [968](#)
 - SupportsMultiThreading, [968](#)
- INetworkArray, [233](#)
 - this[int index], [234](#)
- INetworkAssetSource< T >, [234](#)
 - Acquire, [235](#)
 - Description, [235](#)
 - IsCompleted, [236](#)
 - Release, [235](#)
 - WaitForResult, [235](#)
- INetworkDictionary, [236](#)
 - Add, [236](#)
- INetworkInput, [237](#)
- INetworkLinkedList, [237](#)
 - Add, [237](#)
- INetworkObjectInitializer, [238](#)
 - InitializeNetworkState, [238](#)
- INetworkObjectProvider, [238](#)
 - AcquirePrefabInstance, [239](#)
 - GetPrefabId, [239](#)
 - Initialize, [240](#)
 - ReleaseInstance, [240](#)
 - Shutdown, [240](#)
- INetworkPrefabSource, [241](#)
 - AssetGuid, [241](#)
- INetworkRunnerCallbacks, [241](#)
 - OnConnectedToServer, [243](#)
 - OnConnectFailed, [243](#)
 - OnConnectRequest, [243](#)
 - OnCustomAuthenticationResponse, [243](#)
 - OnDisconnectedFromServer, [244](#)
 - OnHostMigration, [244](#)
 - OnInput, [244](#)
 - OnInputMissing, [244](#)
 - OnObjectEnterAOI, [245](#)
 - OnObjectExitAOI, [245](#)
 - OnPlayerJoined, [245](#)
 - OnPlayerLeft, [245](#)
 - OnReliableDataProgress, [246](#)
 - OnReliableDataReceived, [246](#)
 - OnSceneLoadDone, [246](#)
 - OnSceneLoadStart, [247](#)
 - OnSessionListUpdated, [247](#)
 - OnShutdown, [247](#)
 - OnUserSimulationMessage, [248](#)
- INetworkRunnerUpdater, [248](#)
 - Initialize, [248](#)
 - Shutdown, [249](#)
- INetworkSceneManager, [249](#)
 - GetSceneRef, [250](#)
 - Initialize, [250](#)
 - IsBusy, [253](#)
 - IsRunnerScene, [251](#)
 - LoadScene, [251](#)
 - MainRunnerScene, [253](#)
 - MakeDontDestroyOnLoad, [251](#)
 - MoveGameObjectToScene, [251](#)
 - OnSceneInfoChanged, [251](#)
 - Shutdown, [252](#)
 - TryGetPhysicsScene2D, [252](#)
 - TryGetPhysicsScene3D, [252](#)
 - UnloadScene, [252](#)
- INetworkStruct, [253](#)
- INetworkTRSPTeleport, [254](#)
 - Teleport, [254](#)
- InInterest
 - HitboxRoot, [212](#)
- Init
 - HeapConfiguration, [188](#)
 - NetworkBehaviour.ChangeDetector, [409](#)
 - NetworkConfiguration, [470](#)
 - TickRate, [1108](#)
 - UnityArraySurrogate< T, ReaderWriter >, [258](#)
 - UnityDictionarySurrogate< TKeyType, TKeyReaderWriter, TValueType, TValueReaderWriter >, [260](#)
 - UnityLinkedListSurrogate< T, ReaderWriter >, [262](#)
 - UnitySurrogateBase, [264](#)
 - UnityValueSurrogate< T, TReaderWriter >, [266](#)
- InitHitboxes
 - HitboxRoot, [209](#)
- Initial
 - Fusion, [51](#)
- Initialize
 - INetSocket, [966](#)
 - INetworkObjectProvider, [240](#)
 - INetworkRunnerUpdater, [248](#)
 - INetworkSceneManager, [250](#)
 - NetPeer, [1041](#), [1042](#)
- InitializeNetworkArray< T >
 - NetworkBehaviourUtils, [444](#)
- InitializeNetworkDictionary< D, K, V >
 - NetworkBehaviourUtils, [444](#)
- InitializeNetworkList< T >

- NetworkBehaviourUtils, 445
- InitializeNetworkState
 - INetworkObjectInitializer, 238
 - NetworkObjectInitializerUnity, 559
- InlineHelpAttribute, 254
 - InlineHelpAttribute, 255
 - ShowTypeHelp, 255
- InObjectUpdates
 - FusionStatisticsSnapshot, 1081
- InPackets
 - FusionStatisticsSnapshot, 1081
 - NetworkObjectStatisticsSnapshot, 1091
- InPlayMode
 - ReadOnlyAttribute, 842
- InProgress
 - Fusion, 50
- INPUT
 - AuthorityMasks, 110
- InputAuthority
 - Fusion, 57
 - NetworkObject, 536
 - NetworkObjectHeader, 555
 - NetworkObjectMeta, 560
- InputAuthorityGained
 - IInputAuthorityGained, 231
- InputAuthorityLost
 - IInputAuthorityLost, 231
- InputCount
 - Simulation, 915
- InputDataWordCount
 - SimulationConfig, 931
- InputInBandwidth
 - FusionStatisticsSnapshot, 1082
- InputOutBandwidth
 - FusionStatisticsSnapshot, 1082
- InputQueue
 - EngineProfiler, 149
- InputQueueCallback
 - EngineProfiler, 153
- InputReceiveDelta
 - FusionStatisticsSnapshot, 1082
- InputRecvDelta
 - EngineProfiler, 149
- InputRecvDeltaCallback
 - EngineProfiler, 154
- InputRecvDeltaDeviation
 - EngineProfiler, 149
- InputRecvDeltaDeviationCallback
 - EngineProfiler, 154
- InputSize
 - EngineProfiler, 150
- InputSizeCallback
 - EngineProfiler, 154
- InputTotalWordCount
 - SimulationConfig, 933
- InputTransferMode
 - SimulationConfig, 932
- InputTransferModes
 - SimulationConfig, 931
- Insert
 - NetConnectionMap, 1034
- Instance
 - NetworkObjectSortKeyComparer, 570
- Instances
 - NetworkRunner, 690
- InstantiateInRunnerScene
 - NetworkRunner, 656
- InstantiateInRunnerScene< T >
 - NetworkRunner, 656, 657
- InsufficientSourceAuthority
 - Fusion, 54, 56
- InsufficientTargetAuthority
 - Fusion, 54
- Int
 - NetworkBehaviourBufferInterpolator, 428
- InterestEnter
 - IInterestEnter, 232
- InterestExit
 - IInterestExit, 232
- InternalOnDestroy
 - NetworkBehaviourUtils, 446
- InternalOnDisable
 - NetworkBehaviourUtils, 446
- InternalOnEnable
 - NetworkBehaviourUtils, 446
- InternalStruct
 - Fusion, 52
- InterpAlpha
 - SimulationInputHeader, 940
- InterpFrom
 - SimulationInputHeader, 940
- Interpolated
 - Fusion, 53
- InterpolatedErrorCorrectionSettings, 267
 - MaxRate, 268
 - MinRate, 268
 - PosBlendEnd, 268
 - PosBlendStart, 268
 - PosMinCorrection, 269
 - PosTeleportDistance, 269
 - RotBlendEnd, 269
 - RotBlendStart, 269
 - RotTeleportRadians, 270
- InterpolationOffset
 - EngineProfiler, 150
 - FusionStatisticsSnapshot, 1082
- InterpolationOffsetCallback
 - EngineProfiler, 154
- InterpolationOffsetDeviation
 - EngineProfiler, 150
- InterpolationOffsetDeviationCallback
 - EngineProfiler, 154
- InterpolationSpeed
 - EngineProfiler, 150
 - FusionStatisticsSnapshot, 1082
- InterpolationSpeedCallback

- EngineProfiler, [154](#)
- InterpTo
 - SimulationInputHeader, [941](#)
- IntsRequiredForBits
 - Maths, [326](#)
- Invalid
 - Fusion, [52](#), [58](#)
 - Fusion.Sockets.Stun, [70](#)
 - PlayerRef, [778](#)
- InvalidArguments
 - Fusion, [59](#)
- InvalidAuthentication
 - Fusion, [59](#)
- InvalidEventCode
 - Fusion.Protocol, [66](#)
- InvalidJoinGameMode
 - Fusion.Protocol, [66](#)
- InvalidJoinMsgType
 - Fusion.Protocol, [66](#)
- InvalidRegion
 - Fusion, [59](#)
- InvalidTickRate
 - TickRate, [1106](#)
- InvalidVersion
 - Versioning, [1157](#)
- InvertY
 - NormalizedRectAttribute, [769](#)
- Invoked
 - Fusion, [54](#)
- InvokeLocal
 - RpcAttribute, [863](#)
- InvokeRenderInBatchMode
 - NetworkProjectConfig, [617](#)
- InvokeRpc
 - NetworkBehaviourUtils, [452](#)
- InvokeSceneLoadDone
 - NetworkRunner, [657](#)
- InvokeSceneLoadStart
 - NetworkRunner, [657](#)
- IPlayerJoined, [270](#)
 - PlayerJoined, [270](#)
- IPlayerLeft, [271](#)
 - PlayerLeft, [271](#)
- IPublicFacingInterface, [1160](#)
- IRemotePrefabCreated, [271](#)
 - RemotePrefabCreated, [272](#)
- Is< T >
 - NetworkInput, [503](#)
- IsAcquired
 - NetworkPrefabTable, [606](#)
- IsActiveOnLoad
 - NetworkLoadSceneParameters, [523](#)
- IsAligned
 - Native, [349](#)
- IsBeingDestroyed
 - NetworkObjectReleaseContext, [568](#)
- IsBusy
 - INetworkSceneManager, [253](#)
- ISceneLoadDone, [272](#)
 - SceneLoadDone, [272](#)
- ISceneLoadStart, [273](#)
 - SceneLoadStart, [273](#)
- IsClient
 - NetworkRunner, [690](#)
 - Simulation, [916](#)
- IsCloudReady
 - NetworkRunner, [691](#)
- IsCompleted
 - INetworkAssetSource< T >, [236](#)
 - NetworkSceneAsyncOp.Awaiter, [711](#)
 - NetworkSpawnOp.Awaiter, [731](#)
- IsConnectedToServer
 - NetworkRunner, [691](#)
- IsCreated
 - NetSocket, [1058](#)
- IsCustom
 - NetworkObjectTypeId, [580](#)
- IsDone
 - IAsyncOperation, [221](#)
 - NetworkSceneAsyncOp, [709](#)
- IsEvenBytes
 - BitStream, [823](#)
- IsFailed
 - NetworkSpawnOp, [729](#)
- IsFirstTick
 - NetworkRunner, [691](#)
 - Simulation, [916](#)
- IsForward
 - NetworkRunner, [691](#)
 - Simulation, [916](#)
- IsGlobal
 - FusionGlobalScriptableObject< T >, [181](#)
- IsGlobalLoaded
 - NetworkProjectConfigAsset, [623](#)
- IsGlobalLoadedInternal
 - FusionGlobalScriptableObject< T >, [181](#)
- IsHitboxActive
 - HitboxRoot, [210](#)
- IsIgnored
 - NetworkObjectFlagsExtensions, [539](#)
- ISimulationEnter, [273](#)
 - SimulationEnter, [274](#)
- ISimulationExit, [274](#)
 - SimulationExit, [274](#)
- IsIndex
 - SceneRef, [886](#)
- IsInList
 - NetBitBufferList, [1001](#)
- IsInputAuthority
 - Simulation, [910](#)
- IsInSimulation
 - NetworkObject, [536](#)
- IsInterestedIn
 - NetworkRunner, [657](#)
 - Simulation, [911](#)
- IsInvokeLocal

- RpclInfo, [869](#)
- IsIPv4
 - NetAddress, [973](#)
- IsIPv6
 - NetAddress, [973](#)
- IsLastTick
 - NetworkRunner, [691](#)
 - Simulation, [916](#)
- IsLocalPhysics2D
 - NetworkLoadSceneParameters, [523](#)
- IsLocalPhysics3D
 - NetworkLoadSceneParameters, [523](#)
- IsLocalPlayerFirstExecution
 - Simulation, [916](#)
- IsLocalSimulationInputAuthority
 - Simulation, [911](#)
- IsLocalSimulationStateAuthority
 - Simulation, [911](#), [912](#)
- IsLocalSimulationStateOrInputSource
 - Simulation, [912](#)
- IsMainTRSP
 - NetworkTRSP, [765](#)
- IsMasterClient
 - PlayerRef, [778](#)
 - Simulation, [917](#)
- IsNested
 - NetworkObject, [536](#)
- IsNestedObject
 - NetworkObjectReleaseContext, [569](#)
- IsNone
 - NetworkObjectNestingKey, [564](#)
 - NetworkObjectTypeId, [580](#)
 - NetworkPrefabId, [590](#)
 - PlayerRef, [778](#)
- IsNothing
 - Mask256, [315](#)
- IsOnEvenByte
 - NetBitBuffer, [997](#)
- IsOpen
 - SessionInfo, [903](#)
 - StartGameArgs, [1073](#)
- IsPath
 - SceneRef, [883](#)
- IsSpawned, [275](#)
 - Spawned, [275](#)
- IsPlayer
 - NetworkRunner, [691](#)
 - Simulation, [917](#)
- IsPlayerValid
 - NetworkRunner, [658](#)
- IsPointerAligned
 - Native, [349](#)
- IsPrefab
 - NetworkObjectTypeId, [580](#)
- IsPrime
 - Primes, [781](#)
- IsProxy
 - NetworkBehaviour, [404](#)
- NetworkObject, [536](#)
- IsQueued
 - NetworkSpawnOp, [729](#)
- IsReadOnly
 - SerializableDictionary< TKey, TValue >, [894](#)
- IsRealPlayer
 - PlayerRef, [778](#)
- IsRelayAddr
 - NetAddress, [973](#)
- IsReserved
 - NetworkId, [500](#)
- IsResimulation
 - NetworkRunner, [692](#)
 - Simulation, [917](#)
- IsResume
 - NetworkObject, [534](#)
 - NetworkRunner, [692](#)
- IsRunnerScene
 - INetworkSceneManager, [251](#)
- IsRunning
 - NetworkRunner, [692](#)
 - Simulation, [917](#)
 - TickTimer, [1119](#)
 - Timer, [1122](#)
- IsSceneAuthority
 - NetworkRunner, [692](#)
- IsSceneManagerBusy
 - NetworkRunner, [692](#)
- IsSceneObject
 - NetworkObjectTypeId, [580](#)
- IsServer
 - NetworkRunner, [692](#)
 - Simulation, [917](#)
- IsSet
 - NetworkButtons, [462](#)
- IsSet< T >
 - NetworkButtons, [463](#)
- IsSharedModeMasterClient
 - NetworkRunner, [693](#)
- IsShutdown
 - NetPeer, [1046](#)
 - NetworkRunner, [693](#)
 - Simulation, [917](#)
- IsSimulationUpdating
 - NetworkRunner, [693](#)
- IsSingleLoad
 - NetworkLoadSceneParameters, [523](#)
- IsSinglePlayer
 - NetworkRunner, [693](#)
 - Simulation, [918](#)
- IsSpawnable
 - NetworkObject, [536](#)
- IsSpawned
 - NetworkSpawnOp, [729](#)
- IsStarting
 - NetworkRunner, [693](#)
- IsStateAuthority
 - Simulation, [913](#)

- IsStruct
 - NetworkObjectTypeId, [580](#)
- IsSynchronous
 - NetworkPrefabAcquireContext, [584](#)
 - NetworkPrefabInfo, [592](#)
- IsTargeted
 - SimulationMessage, [945](#)
- IStateAuthorityChanged, [275](#)
 - StateAuthorityChanged, [276](#)
- IsUnreliable
 - SimulationMessage, [953](#)
- IsValid
 - LobbyInfo, [312](#)
 - NetAddress, [973](#)
 - NetConnectionMap.Iterator, [1038](#)
 - NetworkBehaviourId, [437](#)
 - NetworkId, [500](#)
 - NetworkInput, [504](#)
 - NetworkObject, [537](#)
 - NetworkObjectGuid, [549](#)
 - NetworkObjectNestingKey, [565](#)
 - NetworkObjectTypeId, [580](#)
 - NetworkPrefabId, [590](#)
 - NetworkPrefabRef, [600](#)
 - NetworkSceneAsyncOp, [709](#)
 - NetworkSceneObjectId, [723](#)
 - SceneRef, [886](#)
 - SerializableType< BaseType >, [898](#)
 - SessionInfo, [903](#)
 - TickRate, [1108](#), [1109](#)
- IsVersionCurrent
 - NetworkObjectFlagsExtensions, [539](#)
- IsVisible
 - SessionInfo, [903](#)
 - StartGameArgs, [1073](#)
- IsZero
 - Fusion, [47](#)
- ItemsPropertyPath
 - SerializableDictionary< TKey, TValue >, [893](#)
- Iterator
 - NetConnectionMap.Iterator, [1037](#)
- IUnitySurrogate, [255](#)
 - Read, [255](#)
 - Write, [256](#)
- IUnityValueSurrogate< T >, [256](#)
 - DataProperty, [257](#)
- JoinSessionLobby
 - NetworkRunner, [658](#)
- KeepNonStateMembers
 - PreserveInPluginAttribute, [780](#)
- Key
 - NetworkRpcStaticWeavedInvokerAttribute, [630](#)
 - NetworkRpcWeavedInvokerAttribute, [631](#)
 - ReliableId, [1060](#)
 - RpcInvokeData, [871](#)
- Keys
 - SerializableDictionary< TKey, TValue >, [894](#)
- KeywordCount
 - NetworkedWeavedDictionaryAttribute, [488](#)
- Kilobytes
 - Fusion, [60](#)
- Kind
 - NetworkObjectTypeId, [580](#)
- Label
 - EditorButtonAttribute, [141](#)
 - FieldEditorButtonAttribute, [160](#)
- LagCompensatedExt, [290](#)
 - SortDistance, [290](#)
 - SortReference, [291](#)
- LagCompensatedHit, [276](#)
 - Collider, [277](#)
 - Collider2D, [278](#)
 - Distance, [278](#)
 - GameObject, [278](#)
 - Hitbox, [278](#)
 - HitboxColliderPosition, [278](#)
 - HitboxColliderRotation, [278](#)
 - Normal, [279](#)
 - operator LagCompensatedHit, [277](#)
 - Point, [279](#)
 - Type, [279](#)
- LagCompensation
 - NetworkProjectConfig, [617](#)
 - NetworkRunner, [693](#)
- LagCompensationDraw, [291](#)
 - BVHDraw, [292](#)
 - GizmosDrawWireCapsule, [291](#)
 - SnapshotHistoryDraw, [292](#)
- LagCompensationSettings, [309](#)
 - CachedStaticCollidersSize, [310](#)
 - Enabled, [310](#)
 - ExpansionFactor, [311](#)
 - HitboxBufferLengthInMs, [310](#)
 - HitboxDefaultCapacity, [310](#)
 - Optimize, [311](#)
- LagCompensationStatisticsSnapshot, [1085](#)
 - AddOnBufferTime, [1086](#)
 - AddOnBVHTime, [1086](#)
 - AdvanceBufferTime, [1086](#)
 - BVHMaxDeep, [1086](#)
 - BVHNodesCount, [1086](#)
 - HitboxesCount, [1086](#)
 - RefitBVHTime, [1087](#)
 - TotalElapsedTime, [1087](#)
 - UpdateBufferTime, [1087](#)
 - UpdateBVHTime, [1087](#)
- LagCompensationUtils.ContactData, [292](#)
 - Normal, [293](#)
 - Penetration, [293](#)
 - Point, [293](#)
- LastChild
 - Fusion, [61](#)
- LastReceiveTick
 - NetworkObject, [537](#)
- Latest

- Fusion, [53](#)
- LatestServerTick
 - NetworkRunner, [694](#)
 - Simulation, [918](#)
- LatestState
 - SimulationConfig, [931](#)
- LayerAttribute, [311](#)
- LayerMask
 - Query, [296](#)
 - QueryParams, [298](#)
- LayerMatrixAttribute, [311](#)
- Legacy
 - HitboxRoot, [209](#)
- Length
 - CapacityAttribute, [118](#)
 - FixedArray< T >, [169](#)
 - NetworkArray< T >, [378](#)
 - NetworkArrayReadOnly< T >, [383](#)
 - NetworkBehaviourBuffer, [422](#)
 - NetworkString< TSize >, [757](#)
 - RaycastQuery, [302](#)
 - RaycastQueryParams, [304](#)
- LengthBits
 - NetBitBuffer, [997](#)
- LengthBytes
 - NetBitBuffer, [998](#)
- Lerp
 - Angle, [93](#)
 - Maths, [326](#), [327](#)
- Less
 - Fusion, [47](#)
- LessOrEqual
 - Fusion, [47](#)
- Load
 - FusionGlobalScriptableObjectSourceAttribute, [186](#)
 - NetworkPrefabTable, [606](#)
- LoadError
 - Fusion, [50](#)
- LoadId
 - NetworkLoadSceneParameters, [522](#)
 - NetworkSceneObjectId, [722](#)
- LoadScene
 - INetworkSceneManager, [251](#)
 - NetworkRunner, [659](#), [660](#)
- LoadSceneMode
 - NetworkLoadSceneParameters, [523](#)
- LoadSceneParameters
 - NetworkLoadSceneParameters, [523](#)
- LobbyInfo, [311](#)
 - IsValid, [312](#)
 - Name, [312](#)
 - NetworkRunner, [694](#)
 - Region, [312](#)
- Local
 - Fusion, [53](#)
- LocalAddress
 - Simulation, [918](#)
- LocalAlpha
 - NetworkRunner, [694](#)
 - Simulation, [918](#)
- LocalConnectionId
 - NetConnection, [1027](#)
- LocalhostIPv4
 - NetAddress, [971](#)
- LocalhostIPv6
 - NetAddress, [972](#)
- LocalInvokeResult
 - RpclInvokeInfo, [872](#)
- LocalPhysicsMode
 - NetworkLoadSceneParameters, [524](#)
- LocalPlayer
 - NetworkRunner, [694](#)
 - Simulation, [918](#)
- LocalPrefabCreated
 - ILocalPrefabCreated, [233](#)
- LocalRenderTime
 - NetworkRunner, [694](#)
- LocalToWorldMatrix
 - ColliderDrawInfo, [288](#)
- LossChanceMax
 - NetworkSimulationConfiguration, [727](#)
- LossChanceMin
 - NetworkSimulationConfiguration, [727](#)
- LossChancePeriod
 - NetworkSimulationConfiguration, [727](#)
- LossChanceShape
 - NetworkSimulationConfiguration, [727](#)
- LossChanceThreshold
 - NetworkSimulationConfiguration, [727](#)
- LossNotifySequences
 - NetConfigSimulation, [1022](#)
- LossNotifySequencesLength
 - NetConfigSimulation, [1023](#)
- LossOscillator
 - NetConfigSimulation, [1023](#)
- Low
 - Fusion, [53](#)
- Lowest
 - Fusion, [53](#)
- MainRunnerScene
 - INetworkSceneManager, [253](#)
- MakeDontDestroyOnLoad
 - INetworkSceneManager, [251](#)
 - NetworkRunner, [660](#)
- MakeInitializer< K, V >
 - NetworkBehaviour, [396](#)
- MakeInitializer< T >
 - NetworkBehaviour, [396](#)
- MakePtr< T >
 - NetworkBehaviour, [396](#)
- MakeRef< T >
 - NetworkBehaviour, [397](#), [398](#)
- MakeSerializableDictionary< K, V >
 - NetworkBehaviourUtils, [447](#)
- Malloc
 - Native, [349](#)

- Malloc< T >
 - Native, [350](#)
- MallocAndClear
 - Native, [350](#)
- MallocAndClear< T >
 - Native, [351](#)
- MallocAndClearArray
 - Native, [351](#)
- MallocAndClearArray< T >
 - Native, [351](#)
- MallocAndClearArrayMin1< T >
 - Native, [352](#)
- MallocAndClearBlock
 - Native, [352–357](#)
- MallocAndClearPtrArray< T >
 - Native, [357](#)
- MallocAndClearPtrArrayMin1< T >
 - Native, [357](#)
- Manager
 - HitboxRoot, [213](#)
- Manual
 - Fusion.Analyzer, [63](#)
- Mask
 - FieldsMask< T >, [163](#)
 - Maths.FastAbs, [334](#)
- Mask256, [312](#)
 - Clear, [314](#)
 - Equals, [314](#)
 - GetBit, [314](#)
 - GetHashCode, [314](#)
 - IsNothing, [315](#)
 - Mask256, [313](#)
 - operator long, [315](#)
 - operator Mask256, [315](#)
 - operator~, [316](#)
 - operator&, [315](#)
 - operator|, [315](#)
 - SetBit, [316](#)
 - this[int i], [316](#)
 - ToString, [316](#)
- MaskBroadcast
 - Fusion, [56](#)
- MaskCulled
 - Fusion, [56](#)
- MaskNotSent
 - Fusion, [56](#)
- MaskSent
 - Fusion, [56](#)
- MaskVersion
 - Fusion, [50](#)
- MASTER_CLIENT_RAW
 - PlayerRef, [777](#)
- MasterClient
 - PlayerRef, [778](#)
 - SimulationRuntimeConfig, [954](#)
- MasterClientObject
 - Fusion, [50](#)
- MatchmakingMode
 - StartGameArgs, [1073](#)
- Maths, [317](#)
 - BitScanReverse, [318–320](#)
 - BitsRequiredForNumber, [320](#)
 - BytesRequiredForBits, [321](#)
 - CeilToInt, [321](#)
 - Clamp, [322, 323](#)
 - Clamp01, [323, 324](#)
 - ClampToByte, [324](#)
 - CosineInterpolate, [325](#)
 - CountSetBits, [325](#)
 - CountUsedBitsMinOne, [325](#)
 - FloorToInt, [326](#)
 - IntsRequiredForBits, [326](#)
 - Lerp, [326, 327](#)
 - MicrosecondsToSeconds, [327](#)
 - MillisecondsToMicroseconds, [328](#)
 - MillisecondsToSeconds, [328](#)
 - Min, [328](#)
 - NextPowerOfTwo, [329](#)
 - PrintBits, [329](#)
 - QuaternionCompress, [329](#)
 - QuaternionDecompress, [330](#)
 - SecondsToMicroseconds, [330](#)
 - SecondsToMilliseconds, [331](#)
 - SizeOfBits< T >, [331](#)
 - ZigZagDecode, [331, 332](#)
 - ZigZagEncode, [332](#)
- Maths.FastAbs, [334](#)
 - Mask, [334](#)
 - Single, [334](#)
 - UInt32, [334](#)
- MAX
 - NetworkRNG, [628](#)
- Max
 - AABB, [281](#)
 - Angle, [93](#)
 - NetConfigSimulationOscillator, [1025](#)
 - RangeExAttribute, [841](#)
- MAX_HITBOXES
 - HitboxRoot, [212](#)
- MAX_INDEX
 - NetworkPrefabId, [589](#)
- MAX_MTU_BITS_PAYLOAD
 - NetPeer, [1045](#)
- MAX_MTU_BYTES_PAYLOAD
 - NetPeer, [1045](#)
- MAX_MTU_BYTES_TOTAL
 - NetPeer, [1045](#)
- MAX_PACKET_BYTES_PAYLOAD
 - NetPeer, [1045](#)
- MAX_PACKET_BYTES_TOTAL
 - NetPeer, [1045](#)
- MAX_PAYLOAD_SIZE
 - SimulationMessage, [951](#)
- MAX_SCENE_OBJECT_INDEX
 - NetworkObjectTypeId, [578](#)
- MaxCcuReached

- Fusion, [59](#)
- MaxConnections
 - NetConfig, [1017](#)
- MaxDepth
 - BVHNodeDrawInfo, [287](#)
- MaxLateInputs
 - TimeSyncConfiguration, [1123](#)
- MaxLateSnapshots
 - TimeSyncConfiguration, [1123](#)
- MaxLength
 - ArrayLengthAttribute, [101](#)
- MaxPayloadSize
 - RpcAttribute, [862](#)
- MaxPlayers
 - SessionInfo, [903](#)
- MaxRate
 - InterpolatedErrorCorrectionSettings, [268](#)
- MaxScenes
 - NetworkSceneInfo, [715](#)
- MaxSize
 - NetworkSerializeMethodAttribute, [724](#)
- MaxStringByteCountAttribute, [335](#)
 - ByteCount, [336](#)
 - Encoding, [336](#)
 - MaxStringByteCountAttribute, [335](#)
- Medium
 - Fusion, [53](#)
- Megabytes
 - Fusion, [60](#)
- MemClear
 - Native, [358](#)
- MemCmp
 - Native, [358](#)
- MemCpy
 - Native, [359](#)
- MemCpyFast
 - Native, [359](#)
- MemMove
 - Native, [359](#)
- MemoryStatisticsSnapshot, [1087](#)
 - BUCKET_COUNT, [1088](#)
 - BucketFreeBlocksCount, [1088](#)
 - BucketFullBlocksCount, [1089](#)
 - BucketUsedBlocksCount, [1089](#)
 - TargetAllocator, [1088](#)
 - TotalFreeBlocks, [1089](#)
- Message
 - ErrorIfAttribute, [157](#)
 - NetworkObjectSpawnException, [571](#)
 - SimulationMessagePtr, [953](#)
 - WarnIfAttribute, [1158](#)
- MessageSize
 - RpcSendResult, [873](#)
- Meta
 - NetworkPrefabAcquireContext, [584](#)
- META_WORD_COUNT
 - NetworkDictionary< K, V >, [479](#)
- META_WORDS
 - NetworkLinkedList< T >, [513](#)
 - NetworkLinkedListReadOnly< T >, [519](#)
- Method
 - RenderAttribute, [858](#)
 - RpcHeader, [867](#)
- MethodName
 - OnChangedRenderAttribute, [770](#)
- MicrosecondsToSeconds
 - Maths, [327](#)
- MillisecondsToMicroseconds
 - Maths, [328](#)
- MillisecondsToSeconds
 - Maths, [328](#)
- MilliSecs
 - Fusion, [60](#)
- Min
 - AABB, [281](#)
 - Angle, [93](#)
 - Maths, [328](#)
 - NetConfigSimulationOscillator, [1025](#)
 - RangeExAttribute, [841](#)
- MinLength
 - ArrayLengthAttribute, [101](#)
- MinRate
 - InterpolatedErrorCorrectionSettings, [268](#)
- Mode
 - DrawIfAttribute, [130](#)
 - NetworkRunner, [694](#)
 - Simulation, [918](#)
- Modes
 - SimulationBehaviourAttribute, [928](#)
- MonitorNetworkObjectStatistics
 - NetworkObjectStatisticsManager, [1090](#)
- MoreToRead
 - NetBitBuffer, [998](#)
- MoveGameObjectToSameScene
 - NetworkRunner, [660](#)
- MoveGameObjectToScene
 - INetworkSceneManager, [251](#)
 - NetworkRunner, [661](#)
- MoveNext
 - FixedArray< T >.Enumerator, [171](#)
 - NetworkArray< T >.Enumerator, [380](#)
 - NetworkBehaviour.ChangeDetector.Enumerator, [411](#)
 - NetworkDictionary< K, V >.Enumerator, [481](#)
 - NetworkLinkedList< T >.Enumerator, [515](#)
 - UTF32Tools.CharEnumerator, [1138](#)
- MoveToRunnerScene
 - NetworkRunner, [661](#)
- MoveToRunnerScene< T >
 - NetworkRunner, [662](#)
- Multiple
 - NetworkProjectConfig, [613](#)
- Multiplier
 - Fusion, [60](#)
- Name
 - DisplayNameAttribute, [126](#)

- LobbyInfo, [312](#)
- NetworkObject, [537](#)
- SessionInfo, [904](#)
- Native, [336](#)
 - ALIGNMENT, [364](#)
 - AlignPointer, [339](#)
 - ArrayClear< T >, [339](#)
 - ArrayCompare< T >, [340](#)
 - ArrayCopy, [340](#)
 - CACHE_LINE_SIZE, [364](#)
 - CopyFromArray< T >, [341](#)
 - CopyToArray< T >, [341](#)
 - DoubleArray< T >, [342](#)
 - DoublePtrArray< T >, [343](#)
 - Empty< T >, [343](#)
 - Expand, [343](#)
 - ExpandArray< T >, [344](#)
 - ExpandPtrArray< T >, [344](#)
 - Free, [345](#)
 - GetAlignment, [345](#)
 - GetAlignment< T >, [346](#)
 - GetFieldOffset, [346](#)
 - GetLengthPrefixedUTF8ByteCount, [346](#)
 - GetMaxAlignment, [347](#), [348](#)
 - IsAligned, [349](#)
 - IsPointerAligned, [349](#)
 - Malloc, [349](#)
 - Malloc< T >, [350](#)
 - MallocAndClear, [350](#)
 - MallocAndClear< T >, [351](#)
 - MallocAndClearArray, [351](#)
 - MallocAndClearArray< T >, [351](#)
 - MallocAndClearArrayMin1< T >, [352](#)
 - MallocAndClearBlock, [352](#)–[357](#)
 - MallocAndClearPtrArray< T >, [357](#)
 - MallocAndClearPtrArrayMin1< T >, [357](#)
 - MemClear, [358](#)
 - MemCmp, [358](#)
 - MemCpy, [359](#)
 - MemCpyFast, [359](#)
 - MemMove, [359](#)
 - ReadLengthPrefixedUTF8, [360](#)
 - ReferenceToPointer< T >, [360](#)
 - RoundBitsUpTo32, [361](#)
 - RoundBitsUpTo64, [361](#)
 - RoundToAlignment, [361](#), [362](#)
 - RoundToMaxAlignment, [362](#)
 - SizeOf, [363](#)
 - WordCount, [363](#)
 - WriteLengthPrefixedUTF8, [364](#)
- NativeSocket
 - NetSocket, [1057](#)
- NATType
 - Fusion.Sockets.Stun, [70](#)
 - NetworkRunner, [695](#)
- NestedComponentUtilities, [364](#)
 - EnsureRootComponentExists< T, TStopOn >, [366](#)
 - FindObjectsOfTypeInOrder< T >, [366](#)
 - FindObjectsOfTypeInOrder< T, TCast >, [368](#)
 - GetNestedComponentInChildren< T, TStopOn >, [369](#)
 - GetNestedComponentInParent< T, TStopOn >, [370](#)
 - GetNestedComponentInParents< T, TStopOn >, [370](#)
 - GetNestedComponentsInChildren< T >, [370](#)
 - GetNestedComponentsInChildren< T, TSearch, TStop >, [371](#)
 - GetNestedComponentsInChildren< T, TStopOn >, [371](#)
 - GetNestedComponentsInParents< T >, [371](#)
 - GetNestedComponentsInParents< T, TStop >, [372](#)
 - GetParentComponent< T >, [372](#)
- NestedObjects
 - NetworkObject, [534](#)
- NestingKey
 - NetworkObjectHeader, [555](#)
 - NetworkObjectMeta, [560](#)
- NestingRoot
 - NetworkObject, [537](#)
 - NetworkObjectHeader, [556](#)
 - NetworkObjectMeta, [561](#)
- NetAddress, [968](#)
 - ActorId, [972](#)
 - Any, [969](#)
 - AnyIPv6, [970](#)
 - CreateFromIpPort, [970](#)
 - Equals, [971](#)
 - FromActorId, [971](#)
 - GetHashCode, [971](#)
 - HasAddress, [972](#)
 - IsIPv4, [973](#)
 - IsIPv6, [973](#)
 - IsRelayAddr, [973](#)
 - IsValid, [973](#)
 - LocalhostIPv4, [971](#)
 - LocalhostIPv6, [972](#)
 - ToString, [972](#)
- NetAddress.EqualityComparer, [973](#)
 - Equals, [974](#)
 - GetHashCode, [974](#)
- NetBitBuffer, [974](#)
 - Address, [996](#)
 - Allocate, [978](#)
 - BytesRemaining, [997](#)
 - CanRead, [979](#)
 - CanWrite, [979](#)
 - CheckBitCount, [979](#)
 - Clear, [980](#)
 - Data, [997](#)
 - Done, [997](#)
 - DoneOrOverflow, [997](#)
 - GetDataPointer, [980](#)
 - GetOffset, [980](#)
 - IsOnEvenByte, [997](#)

- LengthBits, 997
- LengthBytes, 998
- MoreToRead, 998
- OffsetBits, 998
- OffsetBitsUnsafe, 998
- OffsetBytes, 998
- Overflow, 998
- PadToByteBoundary, 980
- PadToByteBoundaryAndGetPtr, 981
- PeekBoolean, 981
- ReadBoolean, 981
- ReadByte, 981
- ReadBytesAligned, 982
- ReadDouble, 982
- ReadInt16, 982
- ReadInt32, 983
- ReadInt32VarLength, 983
- ReadInt64, 984
- ReadInt64VarLength, 984
- ReadSingle, 984
- ReadString, 985
- ReadUInt16, 985
- ReadUInt32, 986
- ReadUInt32VarLength, 986
- ReadUInt64, 987
- ReadUInt64VarLength, 987
- Release, 987
- ReleaseRef, 988
- ReplaceDataFromBlockWithTemp, 988
- SeekToByteBoundary, 988
- SetBufferLengthBytes, 988
- Write, 989
- WriteBoolean, 989
- WriteByte, 989
- WriteBytesAligned, 990
- WriteDouble, 990
- WriteInt16, 991
- WriteInt32, 991
- WriteInt32AtOffset, 991
- WriteInt32VarLength, 992
- WriteInt64, 992
- WriteInt64VarLength, 993
- WriteSingle, 993
- WriteSlow, 993
- WriteString, 994
- WriteUInt16, 994
- WriteUInt32, 994
- WriteUInt32VarLength, 995
- WriteUInt64, 995
- WriteUInt64AtOffset, 996
- WriteUInt64VarLength, 996
- NetBitBuffer.Offset, 999
 - GetLength, 999
 - Offset, 999
- NetBitBufferList, 1000
 - AddFirst, 1000
 - AddLast, 1001
 - IsInList, 1001
 - Remove, 1001
 - RemoveHead, 1002
- NetBitBufferNull, 1002
 - OffsetBits, 1006
 - PadToByteBoundary, 1003
 - WriteBoolean, 1003
 - WriteByte, 1003
 - WriteBytesAligned, 1004
 - WriteInt32, 1004
 - WriteInt32VarLength, 1004, 1005
 - WriteUInt32VarLength, 1005
 - WriteUInt64VarLength, 1006
- NetBitBufferSerializer, 1006
 - Buffer, 1010
 - Check, 1007
 - Reader, 1008
 - Reading, 1011
 - Serialize, 1008–1010
 - Writer, 1010
 - Writing, 1011
- NetCommandAccepted, 1011
- NetCommandConnect, 1011
- NetCommandDisconnect, 1012
- NetCommandHeader, 1013
 - Create, 1013
- NetCommandRefused, 1014
- NetCommands
 - Fusion.Sockets, 68
- NetConfig, 1014
 - Address, 1016
 - ConnectAttempts, 1016
 - ConnectInterval, 1016
 - ConnectionDefaultRtt, 1016
 - ConnectionGroups, 1016
 - ConnectionPingInterval, 1016
 - ConnectionSendBuffers, 1017
 - ConnectionShutdownTime, 1017
 - ConnectionsPerGroup, 1018
 - ConnectionTimeout, 1017
 - Defaults, 1018
 - MaxConnections, 1017
 - Notify, 1017
 - OperationExpireTime, 1017
 - PacketSize, 1018
 - PacketSizeInBits, 1019
 - Simulation, 1018
 - SocketRecvBuffer, 1018
 - SocketSendBuffer, 1018
- NetConfigNotify, 1019
 - AckForceCount, 1020
 - AckForceTimeout, 1020
 - AckMaskBits, 1020
 - AckMaskBytes, 1020
 - Defaults, 1021
 - SequenceBounds, 1021
 - SequenceBytes, 1020
 - WindowSize, 1020
- NetConfigPointer

- Simulation, 919
- NetConfigSimulation, 1021
 - Defaults, 1023
 - DelayOscillator, 1022
 - DuplicateChance, 1022
 - LossNotifySequences, 1022
 - LossNotifySequencesLength, 1023
 - LossOscillator, 1023
 - WithLossNotifySequences, 1022
- NetConfigSimulationOscillator, 1023
 - Additional, 1025
 - GetCurveValue, 1024
 - Max, 1025
 - Min, 1025
 - Noise, 1024
 - Period, 1025
 - ReverseSaw, 1024
 - Saw, 1024
 - Shape, 1025
 - Sine, 1024
 - Square, 1024
 - Threshold, 1026
 - Triangle, 1024
 - WaveShape, 1024
- NetConnectFailedReason
 - Fusion.Sockets, 68
- NetConnection, 1026
 - Active, 1027
 - ConnectionStatus, 1027
 - LocalConnectionId, 1027
 - RemoteAddress, 1027
 - RemoteConnectionId, 1027
 - RoundTripTime, 1027
 - ToString, 1026
- NetConnectionId, 1028
 - Equals, 1029
 - GetHashCode, 1029
 - Group, 1030
 - GroupIndex, 1030
 - operator!=, 1029
 - operator==, 1029
- NetConnectionId.EqualityComparer, 1030
 - Equals, 1030
 - GetHashCode, 1030
- NetConnectionMap, 1031
 - Allocate, 1032
 - ConnectionsBuffer, 1036
 - Count, 1036
 - CountUsed, 1036
 - Dispose, 1033
 - EntryState, 1032
 - Find, 1033, 1034
 - FindByIndex, 1034
 - Free, 1032
 - Full, 1036
 - Insert, 1034
 - None, 1032
 - Remap, 1035
 - Remove, 1035
 - TryFindByIndex, 1035
 - Used, 1032
- NetConnectionMap.Iterator, 1037
 - Current, 1038
 - IsValid, 1038
 - Iterator, 1037
 - Next, 1038
- NetConnectionStatus
 - Fusion.Sockets, 69
- NetDisconnectReason
 - Fusion.Sockets, 69
- NetPacketType
 - Fusion.Sockets, 69
- NetPeer, 1038
 - Address, 1046
 - Config, 1046
 - DEFAULT_HEADERS, 1045
 - Destroy, 1040
 - GetConfigPointer, 1041
 - GetGroup, 1041
 - GroupCount, 1046
 - Initialize, 1041, 1042
 - IsShutdown, 1046
 - MAX_MTU_BITS_PAYLOAD, 1045
 - MAX_MTU_BYTES_PAYLOAD, 1045
 - MAX_MTU_BYTES_TOTAL, 1045
 - MAX_PACKET_BYTES_PAYLOAD, 1045
 - MAX_PACKET_BYTES_TOTAL, 1045
 - Recv, 1042
 - RemapAddress, 1043
 - Send, 1043
 - Update, 1044
- NetPeerGroup, 1046
 - ChangeConnectionAddressDuringConnecting, 1049
 - Connect, 1049
 - ConnectionCount, 1055
 - ConnectionIterator, 1050
 - Disconnect, 1050
 - GetConnection, 1051
 - GetConnectionByIndex, 1051
 - GetConnectionIdleTime, 1051
 - GetNotifyDataBuffer, 1052
 - GetUnreliableDataBuffer, 1052
 - Group, 1055
 - SendNotifyDataBuffer, 1053
 - SendReliable, 1053
 - SendUnconnectedData, 1053
 - SendUnreliableDataBuffer, 1054
 - Time, 1055
 - TryGetConnectionByIndex, 1054
 - Update, 1055
- NetSendEnvelope, 1056
 - SendTime, 1056
 - Sequence, 1056
 - UserData, 1056
- NetSocket, 1057

- Handle, [1057](#)
- IsCreated, [1058](#)
- NativeSocket, [1057](#)
- Network
 - NetworkProjectConfig, [618](#)
- NetworkArray
 - NetworkArray< T >, [374](#)
- NetworkArray< T >, [372](#)
 - Clear, [374](#)
 - CopyFrom, [375](#)
 - CopyTo, [376](#)
 - Get, [377](#)
 - GetEnumerator, [377](#)
 - Length, [378](#)
 - NetworkArray, [374](#)
 - operator NetworkArrayReadOnly< T >, [377](#)
 - Set, [377](#)
 - this[int index], [378](#)
 - ToArray, [377](#)
 - ToListString, [378](#)
 - ToReadOnly, [378](#)
 - ToString, [378](#)
- NetworkArray< T >.Enumerator, [379](#)
 - Current, [380](#)
 - Dispose, [380](#)
 - Enumerator, [379](#)
 - MoveNext, [380](#)
 - Reset, [380](#)
- NetworkArrayExtensions, [381](#)
 - GetRef< T >, [381](#)
 - IndexOf< T >, [381](#)
- NetworkArrayReadOnly< T >, [382](#)
 - Length, [383](#)
 - this[int index], [383](#)
- NetworkAssemblyIgnoreAttribute, [383](#)
- NetworkAssemblyWeavedAttribute, [383](#)
- NetworkBehaviour, [383](#)
 - ChangedTick, [403](#)
 - CopyBackingFieldsToState, [387](#)
 - CopyStateFrom, [387](#)
 - CopyStateToBackingFields, [387](#)
 - Despawned, [387](#)
 - DynamicWordCount, [404](#)
 - FixedUpdateNetwork, [388](#)
 - GetArrayReader< T >, [388](#)
 - GetBehaviourReader< T >, [389](#), [390](#)
 - GetBehaviourReader< TBehaviour, TProperty >, [390](#)
 - GetChangeDetector, [391](#)
 - GetDictionaryReader< K, V >, [391](#), [392](#)
 - GetInput< T >, [392](#)
 - GetLinkedListReader< T >, [393](#)
 - GetLocalAuthorityMask, [394](#)
 - GetPropertyReader< T >, [394](#)
 - GetPropertyReader< TBehaviour, TProperty >, [395](#)
 - HasInputAuthority, [404](#)
 - HasStateAuthority, [404](#)
 - Id, [404](#)
 - IsProxy, [404](#)
 - MakeInitializer< K, V >, [396](#)
 - MakeInitializer< T >, [396](#)
 - MakePtr< T >, [396](#)
 - MakeRef< T >, [397](#), [398](#)
 - NetworkDeserialize, [398](#)
 - NetworkSerialize, [399](#)
 - NetworkUnwrap, [399](#)
 - NetworkWrap, [400](#)
 - offset, [403](#)
 - operator NetworkBehaviourId, [400](#)
 - ReinterpretState< T >, [401](#)
 - ReplicateTo, [401](#), [402](#)
 - ReplicateToAll, [402](#)
 - ResetState, [402](#)
 - Spawned, [402](#)
 - StateBuffer, [404](#)
 - StateBufferIsValid, [405](#)
 - TryGetSnapshotsBuffers, [403](#)
- NetworkBehaviour.ArrayReader< T >, [405](#)
 - Read, [405](#)
- NetworkBehaviour.BehaviourReader< T >, [406](#)
 - Read, [406](#)
 - T, [407](#)
- NetworkBehaviour.ChangeDetector, [407](#)
 - DetectChanges, [408](#), [409](#)
 - Init, [409](#)
 - SimulationState, [408](#)
 - SnapshotFrom, [408](#)
 - SnapshotTo, [408](#)
 - Source, [408](#)
- NetworkBehaviour.ChangeDetector.Enumerable, [410](#)
 - Changed, [410](#)
 - GetEnumerator, [410](#)
- NetworkBehaviour.ChangeDetector.Enumerator, [411](#)
 - Current, [412](#)
 - MoveNext, [411](#)
 - Reset, [411](#)
- NetworkBehaviour.DictionaryReader< K, V >, [412](#)
 - Read, [412](#)
- NetworkBehaviour.LinkedListReader< T >, [413](#)
 - Read, [413](#)
- NetworkBehaviour.PropertyReader< T >, [414](#)
 - PropertyReader, [414](#)
 - Read, [415](#)
 - T, [415](#)
- NetworkBehaviourBuffer, [415](#)
 - Length, [422](#)
 - operator bool, [416](#)
 - Read, [417](#), [419](#)
 - Read< T >, [419](#), [421](#)
 - ReinterpretState< T >, [421](#)
 - this[int index], [422](#)
 - Tick, [422](#)
 - Valid, [423](#)
- NetworkBehaviourBufferInterpolator, [423](#)
 - Alpha, [433](#)

- Angle, [425](#)
- Behaviour, [433](#)
- Bool, [425](#), [426](#)
- Float, [426](#)
- From, [433](#)
- Int, [428](#)
- NetworkBehaviourBufferInterpolator, [424](#)
- operator bool, [428](#)
- Quaternion, [429](#)
- Select< T >, [429](#), [430](#)
- To, [433](#)
- Valid, [433](#)
- Vector2, [430](#), [431](#)
- Vector3, [431](#), [432](#)
- Vector4, [432](#)
- NetworkBehaviourId, [434](#)
 - Behaviour, [437](#)
 - Equals, [435](#)
 - GetHashCode, [435](#)
 - IsValid, [437](#)
 - None, [437](#)
 - Object, [437](#)
 - operator!=, [436](#)
 - operator==, [436](#)
 - SIZE, [437](#)
 - ToString, [436](#)
- NetworkBehaviourUtils, [438](#)
 - CloneArray< T >, [440](#)
 - CopyFromNetworkArray< T >, [440](#)
 - CopyFromNetworkDictionary< D, K, V >, [441](#)
 - CopyFromNetworkList< T >, [441](#)
 - GetMetaData, [442](#)
 - GetRpcStaticIndexOrThrow, [442](#)
 - GetStaticWordCount, [443](#)
 - GetWordCount, [443](#)
 - HasStaticWordCount, [443](#)
 - InitializeNetworkArray< T >, [444](#)
 - InitializeNetworkDictionary< D, K, V >, [444](#)
 - InitializeNetworkList< T >, [445](#)
 - InternalOnDestroy, [446](#)
 - InternalOnDisable, [446](#)
 - InternalOnEnable, [446](#)
 - InvokeRpc, [452](#)
 - MakeSerializableDictionary< K, V >, [447](#)
 - NotifyLocalSimulationNotAllowedToSendRpc, [447](#)
 - NotifyLocalTargetedRpcCulled, [448](#)
 - NotifyNetworkUnwrapFailed< T >, [448](#)
 - NotifyNetworkWrapFailed< T >, [448](#), [449](#)
 - NotifyRpcPayloadSizeExceeded, [449](#)
 - NotifyRpcTargetUnreachable, [449](#)
 - RegisterMetaData, [450](#)
 - RegisterRpcInvokeDelegates, [450](#)
 - ShouldRegisterRpcInvokeDelegates, [450](#)
 - ThrowIfBehaviourNotInitialized, [451](#)
 - TryGetRpcInvokeDelegateArray, [451](#)
 - TryGetRpcStaticInvokeDelegate, [451](#)
- NetworkBehaviourUtils.ArrayInitializer< T >, [452](#)
 - operator NetworkArray< T >, [453](#)
 - operator NetworkLinkedList< T >, [453](#)
- NetworkBehaviourUtils.DictionaryInitializer< K, V >, [454](#)
 - operator NetworkDictionary< K, V >, [454](#)
- NetworkBehaviourUtils.MetaData, [455](#)
- NetworkBehaviourWeavedAttribute, [455](#)
 - NetworkBehaviourWeavedAttribute, [455](#)
 - WordCount, [456](#)
- NetworkBool, [456](#)
 - Equals, [457](#)
 - GetHashCode, [458](#)
 - NetworkBool, [457](#)
 - operator bool, [458](#)
 - operator NetworkBool, [458](#)
 - SIZE, [459](#)
 - ToString, [459](#)
- NetworkButtons, [459](#)
 - Bits, [468](#)
 - Equals, [461](#)
 - GetHashCode, [461](#)
 - GetPressed, [462](#)
 - GetReleased, [462](#)
 - IsSet, [462](#)
 - IsSet< T >, [463](#)
 - NetworkButtons, [460](#), [468](#)
 - Set, [463](#)
 - Set< T >, [464](#)
 - SetAllDown, [464](#)
 - SetAllUp, [464](#)
 - SetDown, [464](#)
 - SetDown< T >, [465](#)
 - SetUp, [465](#)
 - SetUp< T >, [465](#)
 - WasPressed, [466](#)
 - WasPressed< T >, [466](#)
 - WasReleased, [467](#)
 - WasReleased< T >, [467](#)
- NetworkConditions
 - NetworkProjectConfig, [618](#)
- NetworkConfiguration, [468](#)
 - ClientToClientWithServerProxy, [470](#)
 - ClientToServer, [470](#)
 - ConnectAttempts, [471](#)
 - ConnectInterval, [471](#)
 - ConnectionDefaultRtt, [471](#)
 - ConnectionPingInterval, [471](#)
 - ConnectionShutdownTime, [470](#)
 - ConnectionTimeout, [470](#)
 - Init, [470](#)
 - ReliableDataTransferModes, [470](#)
 - ReliableDataTransfers, [469](#)
 - SocketRecvBufferSize, [471](#)
 - SocketSendBufferSize, [472](#)
- NetworkConnected
 - Simulation, [913](#)
- NetworkDelegates, [472](#)
- NetworkDeserialize
 - NetworkBehaviour, [398](#)

- NetworkDeserializeMethodAttribute, 473
- NetworkDictionary
 - NetworkDictionary< K, V >, 475
- NetworkDictionary< K, V >, 473
 - Add, 476
 - Capacity, 479
 - Clear, 476
 - ContainsKey, 476
 - ContainsValue, 476
 - Count, 479
 - Get, 477
 - GetEnumerator, 477
 - META_WORD_COUNT, 479
 - NetworkDictionary, 475
 - operator NetworkDictionaryReadOnly< K, V >, 477
 - Remove, 477, 478
 - Set, 478
 - this[K key], 479
 - ToReadOnly, 478
 - TryGet, 478
- NetworkDictionary< K, V >.Enumerator, 480
 - Current, 481
 - Dispose, 480
 - MoveNext, 481
 - Reset, 481
- NetworkDictionaryReadOnly< K, V >, 481
 - Capacity, 483
 - Count, 483
 - Get, 483
 - TryGet, 483
- NetworkDisconnected
 - Simulation, 914
- NetworkedAttribute, 484
 - Default, 484
 - NetworkedAttribute, 484
- NetworkedBehaviours
 - NetworkObject, 534
- NetworkedWeavedArrayAttribute, 485
 - Capacity, 485
 - ElementReaderWriterType, 485
 - ElementWordCount, 486
- NetworkedWeavedAttribute, 486
 - NetworkedWeavedAttribute, 486
 - WordCount, 487
 - WordOffset, 487
- NetworkedWeavedDictionaryAttribute, 487
 - Capacity, 488
 - KeywordCount, 488
 - ValueWordCount, 488
- NetworkedWeavedLinkedListAttribute, 488
 - Capacity, 489
 - ElementReaderWriterType, 489
 - ElementWordCount, 489
- NetworkedWeavedStringAttribute, 1160
 - CacheFieldName, 1161
 - Capacity, 1161
 - NetworkedWeavedStringAttribute, 1160
- NetworkEvents, 489
 - NetworkEvents.ConnectFailedEvent, 491
 - NetworkEvents.ConnectRequestEvent, 491
 - NetworkEvents.CustomAuthenticationResponse, 491
 - NetworkEvents.DisconnectFromServerEvent, 491
 - NetworkEvents.HostMigrationEvent, 492
 - NetworkEvents.InputEvent, 492
 - NetworkEvents.InputPlayerEvent, 492
 - NetworkEvents.ObjectEvent, 492
 - NetworkEvents.ObjectPlayerEvent, 493
 - NetworkEvents.PlayerEvent, 493
 - NetworkEvents.ReliableDataEvent, 493
 - NetworkEvents.ReliableProgressEvent, 493
 - NetworkEvents.RunnerEvent, 494
 - NetworkEvents.SessionListUpdateEvent, 494
 - NetworkEvents.ShutdownEvent, 494
 - NetworkEvents.SimulationMessageEvent, 494
- NetworkId, 495
 - ALIGNMENT, 499
 - BLOCK_SIZE, 499
 - Comparer, 500
 - CompareTo, 496
 - Equals, 497
 - GetHashCode, 497
 - IsReserved, 500
 - IsValid, 500
 - operator bool, 497
 - operator!=, 498
 - operator==, 498
 - Raw, 500
 - Read, 498
 - SIZE, 500
 - ToNamePrefixString, 498
 - ToString, 499
 - Write, 499
- NetworkId.EqualityComparer, 501
 - Equals, 501
 - GetHashCode, 501
- NetworkIdsObjectName
 - NetworkProjectConfig, 618
- NetworkInput, 501
 - Convert< T >, 502
 - Data, 504
 - Get< T >, 502
 - Is< T >, 503
 - IsValid, 504
 - Set< T >, 503
 - TryGet< T >, 503
 - TrySet< T >, 503
 - Type, 504
 - WordCount, 504
- NetworkInputUtils, 505
 - GetMaxWordCount, 505
 - GetType, 505
 - GetTypeKey, 506
 - GetWordCount, 506
- NetworkInputWeavedAttribute, 506
 - NetworkInputWeavedAttribute, 507

- WordCount, [507](#)
- NetworkLinkedList
 - NetworkLinkedList< T >, [509](#)
- NetworkLinkedList< T >, [507](#)
 - Add, [510](#)
 - Capacity, [513](#)
 - Clear, [510](#)
 - Contains, [511](#)
 - Count, [513](#)
 - ELEMENT_WORDS, [513](#)
 - Get, [511](#)
 - GetEnumerator, [511](#)
 - IndexOf, [511](#)
 - META_WORDS, [513](#)
 - NetworkLinkedList, [509](#)
 - Remap, [512](#)
 - Remove, [512](#)
 - Set, [512](#)
 - this[int index], [513](#)
- NetworkLinkedList< T >.Enumerator, [514](#)
 - Current, [515](#)
 - Dispose, [514](#)
 - MoveNext, [515](#)
 - Reset, [515](#)
- NetworkLinkedListReadOnly< T >, [516](#)
 - Capacity, [519](#)
 - Contains, [518](#)
 - Count, [519](#)
 - ELEMENT_WORDS, [519](#)
 - Get, [518](#)
 - IndexOf, [518](#)
 - META_WORDS, [519](#)
 - this[int index], [519](#)
- NetworkLoadSceneParameters, [520](#)
 - Equals, [521](#)
 - GetHashCode, [521](#)
 - IsActiveOnLoad, [523](#)
 - IsLocalPhysics2D, [523](#)
 - IsLocalPhysics3D, [523](#)
 - IsSingleLoad, [523](#)
 - LoadId, [522](#)
 - LoadSceneMode, [523](#)
 - LoadSceneParameters, [523](#)
 - LocalPhysicsMode, [524](#)
 - operator!=, [521](#)
 - operator==, [522](#)
 - ToString, [522](#)
- NetworkMecanimAnimator, [524](#)
 - Animator, [526](#)
 - ApplyTiming, [526](#)
 - DynamicWordCount, [526](#)
 - SetTrigger, [525](#)
- NetworkObject, [527](#)
 - AssignInputAuthority, [529](#)
 - Awake, [529](#)
 - CopyStateFrom, [529](#), [530](#)
 - Flags, [534](#)
 - ForceRemoteRenderTimeframe, [534](#)
 - GetLocalAuthorityMask, [530](#)
 - GetWordCount, [530](#)
 - HasInputAuthority, [535](#)
 - HasStateAuthority, [535](#)
 - Id, [536](#)
 - InputAuthority, [536](#)
 - IsInSimulation, [536](#)
 - IsNested, [536](#)
 - IsProxy, [536](#)
 - IsResume, [534](#)
 - IsSpawnable, [536](#)
 - IsValid, [537](#)
 - LastReceiveTick, [537](#)
 - Name, [537](#)
 - NestedObjects, [534](#)
 - NestingRoot, [537](#)
 - NetworkedBehaviours, [534](#)
 - NetworkTypeId, [534](#)
 - NetworkUnwrap, [531](#)
 - NetworkWrap, [531](#)
 - OnDestroy, [532](#)
 - operator NetworkId, [532](#)
 - PriorityCallback, [535](#)
 - PriorityLevelDelegate, [532](#)
 - ReleaseStateAuthority, [532](#)
 - RemoveInputAuthority, [533](#)
 - RenderSource, [537](#)
 - RenderTime, [537](#)
 - RenderTimeframe, [538](#)
 - ReplicateTo, [535](#)
 - ReplicateToDelegate, [533](#)
 - RequestStateAuthority, [533](#)
 - Runner, [538](#)
 - SetPlayerAlwaysInterested, [533](#)
 - SortKey, [535](#)
 - StateAuthority, [538](#)
- NetworkObjectAcquireResult
 - Fusion, [49](#)
- NetworkObjectFlags
 - Fusion, [49](#)
- NetworkObjectFlagsExtensions, [538](#)
 - GetVersion, [539](#)
 - IsIgnored, [539](#)
 - IsVersionCurrent, [539](#)
 - SetCurrentVersion, [540](#)
 - SetIgnored, [540](#)
- NetworkObjectGuid, [540](#)
 - ALIGNMENT, [548](#)
 - CompareTo, [544](#)
 - Empty, [549](#)
 - Equals, [545](#)
 - GetHashCode, [545](#)
 - IsValid, [549](#)
 - NetworkObjectGuid, [542](#), [544](#)
 - operator Guid, [545](#)
 - operator NetworkObjectGuid, [546](#)
 - operator NetworkPrefabRef, [546](#)
 - operator!=, [547](#)

- operator==, [547](#)
- Parse, [547](#)
- RawGuidValue, [549](#)
- SIZE, [549](#)
- ToString, [547](#), [548](#)
- ToUnityGuidString, [548](#)
- TryParse, [548](#)
- NetworkObjectGuid.EqualityComparer, [549](#)
 - Equals, [550](#)
 - GetHashCode, [550](#)
- NetworkObjectHeader, [550](#)
 - _reserved, [555](#)
 - BehaviourCount, [555](#)
 - ByteCount, [557](#)
 - Equals, [552](#)
 - Flags, [555](#)
 - GetBehaviourChangedTickArray, [552](#)
 - GetDataPointer, [552](#)
 - GetDataWordCount, [553](#)
 - GetHashCode, [553](#)
 - GetMainNetworkTRSPData, [553](#)
 - HasMainNetworkTRSP, [554](#)
 - Id, [555](#)
 - InputAuthority, [555](#)
 - NestingKey, [555](#)
 - NestingRoot, [556](#)
 - operator!=, [554](#)
 - operator==, [554](#)
 - PLAYER_DATA_WORD, [556](#)
 - SIZE, [556](#)
 - StateAuthority, [556](#)
 - ToString, [554](#)
 - Type, [556](#)
 - WordCount, [556](#)
 - WORDS, [557](#)
- NetworkObjectHeaderFlags
 - Fusion, [50](#)
- NetworkObjectHeaderPtr, [557](#)
 - Id, [558](#)
 - operator NetworkObjectHeaderPtr, [558](#)
 - Ptr, [558](#)
 - Type, [558](#)
- NetworkObjectInitializerUnity, [559](#)
 - InitializeNetworkState, [559](#)
- NetworkObjectMeta, [559](#)
 - Flags, [560](#)
 - Id, [560](#)
 - InputAuthority, [560](#)
 - NestingKey, [560](#)
 - NestingRoot, [561](#)
 - StateAuthority, [561](#)
 - Type, [561](#)
- NetworkObjectNestingKey, [561](#)
 - ALIGNMENT, [564](#)
 - Equals, [563](#)
 - GetHashCode, [563](#)
 - IsNone, [564](#)
 - IsValid, [565](#)
 - NetworkObjectNestingKey, [562](#)
 - SIZE, [564](#)
 - ToString, [564](#)
 - Value, [564](#)
- NetworkObjectNestingKey.EqualityComparer, [565](#)
 - Equals, [565](#)
 - GetHashCode, [566](#)
- NetworkObjectPrefabData, [566](#)
 - Guid, [566](#)
- NetworkObjectProviderDummy, [567](#)
- NetworkObjectReleaseContext, [567](#)
 - IsBeingDestroyed, [568](#)
 - IsNestedObject, [569](#)
 - NetworkObjectReleaseContext, [568](#)
 - Object, [569](#)
 - ToString, [568](#)
 - TypeId, [569](#)
- NetworkObjectSortKeyComparer, [569](#)
 - Compare, [570](#)
 - Instance, [570](#)
- NetworkObjectSpawnDelegate
 - Fusion, [61](#)
- NetworkObjectSpawnException, [570](#)
 - Message, [571](#)
 - NetworkObjectSpawnException, [571](#)
 - Status, [571](#)
 - TypeId, [571](#)
- NetworkObjectStatisticsManager, [1089](#)
 - ClearMonitoredNetworkObjects, [1090](#)
 - GetNetworkObjectStatistics, [1090](#)
 - MonitorNetworkObjectStatistics, [1090](#)
- NetworkObjectStatisticsSnapshot, [1090](#)
 - InBandwidth, [1091](#)
 - InPackets, [1091](#)
 - OutBandwidth, [1091](#)
 - OutPackets, [1091](#)
- NetworkObjectTypeId, [572](#)
 - _value0, [577](#)
 - _value1, [577](#)
 - ALIGNMENT, [578](#)
 - AsCustom, [578](#)
 - AsInternalStructId, [579](#)
 - AsPrefabId, [579](#)
 - AsSceneObjectId, [579](#)
 - Comparer, [579](#)
 - Equals, [574](#)
 - FromCustom, [574](#)
 - FromPrefabId, [574](#)
 - FromSceneObjectId, [575](#)
 - FromSceneRefAndObjectIndex, [575](#)
 - FromStruct, [576](#)
 - GetHashCode, [576](#)
 - IsCustom, [580](#)
 - IsNone, [580](#)
 - IsPrefab, [580](#)
 - IsSceneObject, [580](#)
 - IsStruct, [580](#)
 - IsValid, [580](#)

- Kind, [580](#)
- MAX_SCENE_OBJECT_INDEX, [578](#)
- operator NetworkObjectTypeId, [576](#)
- operator!=, [577](#)
- operator==, [577](#)
- PlayerData, [581](#)
- SIZE, [578](#)
- ToString, [577](#)
- NetworkObjectTypeId.EqualityComparer, [581](#)
 - Equals, [581](#)
 - GetHashCode, [581](#)
- NetworkPhysicsInfo, [582](#)
 - SIZE, [582](#)
 - TimeScale, [582](#)
 - WORD_COUNT, [582](#)
- NetworkPrefabAcquireContext, [583](#)
 - Data, [584](#)
 - DontDestroyOnLoad, [584](#)
 - HasHeader, [585](#)
 - IsSynchronous, [584](#)
 - Meta, [584](#)
 - NetworkPrefabAcquireContext, [583](#)
 - PrefabId, [584](#)
- NetworkPrefabAttribute, [585](#)
- NetworkPrefabId, [585](#)
 - ALIGNMENT, [589](#)
 - AsIndex, [590](#)
 - CompareTo, [587](#)
 - Equals, [587](#)
 - FromIndex, [587](#)
 - FromRaw, [588](#)
 - GetHashCode, [588](#)
 - IsNone, [590](#)
 - IsValid, [590](#)
 - MAX_INDEX, [589](#)
 - operator!=, [588](#)
 - operator==, [588](#)
 - RawValue, [589](#)
 - SIZE, [589](#)
 - ToString, [588](#), [589](#)
- NetworkPrefabId.EqualityComparer, [590](#)
 - Equals, [591](#)
 - GetHashCode, [591](#)
- NetworkPrefabInfo, [591](#)
 - Data, [592](#)
 - HasHeader, [592](#)
 - Header, [592](#)
 - IsSynchronous, [592](#)
 - Prefab, [592](#)
- NetworkPrefabRef, [593](#)
 - ALIGNMENT, [599](#)
 - CompareTo, [595](#)
 - Empty, [600](#)
 - Equals, [596](#)
 - GetHashCode, [596](#)
 - IsValid, [600](#)
 - NetworkPrefabRef, [594](#), [595](#)
 - operator Guid, [597](#)
 - operator NetworkObjectGuid, [597](#)
 - operator NetworkPrefabRef, [597](#)
 - operator!=, [598](#)
 - operator==, [598](#)
 - Parse, [598](#)
 - RawGuidValue, [600](#)
 - SIZE, [600](#)
 - ToString, [598](#), [599](#)
 - ToUnityGuidString, [599](#)
 - TryParse, [599](#)
- NetworkPrefabRef.EqualityComparer, [600](#)
 - Equals, [601](#)
 - GetHashCode, [601](#)
- NetworkPrefabTable, [601](#)
 - AddInstance, [603](#)
 - AddSource, [603](#)
 - Clear, [603](#)
 - Contains, [604](#)
 - GetEntries, [604](#)
 - GetGuid, [604](#)
 - GetId, [604](#)
 - GetInstancesCount, [605](#)
 - GetSource, [605](#), [606](#)
 - IsAcquired, [606](#)
 - Load, [606](#)
 - Options, [609](#)
 - Prefabs, [609](#)
 - RemoveInstance, [607](#)
 - TryAddSource, [607](#)
 - Unload, [608](#)
 - UnloadAll, [608](#)
 - UnloadUnreferenced, [608](#)
 - Version, [609](#)
- NetworkPrefabTableGetPrefabResult
 - Fusion, [50](#)
- NetworkPrefabTableOptions, [609](#)
 - Default, [610](#)
 - UnloadPrefabOnReleasingLastInstance, [610](#)
 - UnloadUnusedPrefabsOnShutdown, [610](#)
- NetworkProjectConfig, [610](#)
 - AllowClientServerModesInWebGL, [615](#)
 - AssembliesToWeave, [615](#)
 - BuildTypes, [615](#)
 - CheckNetworkedPropertiesBeingEmpty, [615](#)
 - CheckRpcAttributeUsage, [616](#)
 - CurrentTypeId, [616](#)
 - CurrentVersion, [616](#)
 - DefaultResourceName, [616](#)
 - Deserialize, [613](#)
 - EncryptionConfig, [616](#)
 - EnqueueIncompleteSynchronousSpawns, [616](#)
 - GetExecutionOrder, [614](#)
 - Global, [620](#)
 - Heap, [617](#)
 - HideNetworkObjectInactivityGuard, [617](#)
 - HostMigration, [617](#)
 - InvokeRenderInBatchMode, [617](#)
 - LagCompensation, [617](#)

- Multiple, [613](#)
- Network, [618](#)
- NetworkConditions, [618](#)
- NetworkIdsObjectName, [618](#)
- None, [613](#)
- NullChecksForNetworkedProperties, [618](#)
- PeerMode, [618](#)
- PeerModes, [613](#)
- PrefabTable, [618](#)
- ReplicationFeatures, [613](#)
- Scheduling, [613](#)
- SchedulingAndInterestManagement, [613](#)
- Serialize, [614](#)
- Simulation, [619](#)
- Single, [613](#)
- TimeSynchronizationOverride, [619](#)
- ToString, [614](#)
- TypeId, [619](#)
- UnloadGlobal, [615](#)
- UseSerializableDictionary, [619](#)
- Version, [619](#)
- NetworkProjectConfigAsset, [620](#)
 - BehaviourMeta, [622](#)
 - Config, [622](#)
 - Global, [622](#)
 - IsGlobalLoaded, [623](#)
 - OnDisable, [621](#)
 - PrefabOptions, [622](#)
 - Prefabs, [622](#)
 - TryGetGlobal, [621](#)
 - UnloadGlobal, [621](#)
- NetworkProjectConfigAsset.SerializableSimulationBehaviourMeta, [623](#)
 - ExecutionOrder, [623](#)
 - Type, [623](#)
- NetworkReceiveDone
 - Simulation, [914](#)
- NetworkRNG, [624](#)
 - MAX, [628](#)
 - NetworkRNG, [625](#)
 - Next, [625](#)
 - NextExclusive, [625](#)
 - NextInt32, [626](#)
 - NextSingle, [626](#)
 - NextSingleExclusive, [626](#)
 - NextUInt32, [626](#)
 - Peek, [629](#)
 - RangeExclusive, [627](#)
 - RangeInclusive, [627](#), [628](#)
 - SIZE, [628](#)
 - ToString, [628](#)
- NetworkRpcStaticWeavedInvokerAttribute, [629](#)
 - Key, [630](#)
 - NetworkRpcStaticWeavedInvokerAttribute, [629](#)
- NetworkRpcWeavedInvokerAttribute, [630](#)
 - Key, [631](#)
 - NetworkRpcWeavedInvokerAttribute, [630](#)
 - Sources, [631](#)
- Targets, [631](#)
- NetworkRunner, [631](#)
 - ActivePlayers, [689](#)
 - AddCallbacks, [641](#)
 - AddGlobal, [642](#)
 - AddPlayerAreaOfInterest, [642](#)
 - Attach, [642](#), [643](#)
 - AuthenticationValues, [689](#)
 - BuildType, [689](#)
 - BuildTypes, [641](#)
 - CanSpawn, [689](#)
 - ClearPlayerAreaOfInterest, [643](#)
 - CloudConnectionLostHandler, [643](#)
 - Config, [690](#)
 - CurrentConnectionType, [690](#)
 - Debug, [641](#)
 - DeltaTime, [690](#)
 - Despawn, [643](#)
 - DestroySingleton< T >, [644](#)
 - Disconnect, [644](#)
 - EnsureRunnerSceneIsActive, [644](#)
 - Exists, [645](#)
 - FindObject, [645](#)
 - GameMode, [690](#)
 - GetAllBehaviours, [646](#)
 - GetAllBehaviours< T >, [646](#)
 - GetAllNetworkObjects, [647](#)
 - GetAreaOfInterestGizmoData, [647](#)
 - GetAvailableRegions, [648](#)
 - GetInputForPlayer< T >, [648](#)
 - GetInstancesEnumerator, [648](#)
 - GetInterfaceListHead, [648](#)
 - GetInterfaceListNext, [650](#)
 - GetInterfaceListPrev, [650](#)
 - GetInterfaceListsCount, [650](#)
 - GetMemorySnapshot, [651](#)
 - GetObjectsInAreaOfInterestForPlayer, [651](#)
 - GetPhysicsScene, [651](#)
 - GetPhysicsScene2D, [652](#)
 - GetPlayerActorId, [652](#)
 - GetPlayerConnectionToken, [652](#)
 - GetPlayerConnectionType, [653](#)
 - GetPlayerObject, [653](#)
 - GetPlayerRtt, [653](#)
 - GetPlayerUserId, [654](#)
 - GetRawInputForPlayer, [654](#)
 - GetResumeSnapshotNetworkObjects, [654](#)
 - GetResumeSnapshotNetworkSceneObjects, [654](#)
 - GetRpcTargetStatus, [655](#)
 - GetRunnerForGameObject, [655](#)
 - GetRunnerForScene, [655](#)
 - GetSingleton< T >, [656](#)
 - HasSingleton< T >, [656](#)
 - Instances, [690](#)
 - InstantiateInRunnerScene, [656](#)
 - InstantiateInRunnerScene< T >, [656](#), [657](#)
 - InvokeSceneLoadDone, [657](#)
 - InvokeSceneLoadStart, [657](#)

- IsClient, [690](#)
- IsCloudReady, [691](#)
- IsConnectedToServer, [691](#)
- IsFirstTick, [691](#)
- IsForward, [691](#)
- IsInterestedIn, [657](#)
- IsLastTick, [691](#)
- IsPlayer, [691](#)
- IsPlayerValid, [658](#)
- IsResimulation, [692](#)
- IsResume, [692](#)
- IsRunning, [692](#)
- IsSceneAuthority, [692](#)
- IsSceneManagerBusy, [692](#)
- IsServer, [692](#)
- IsSharedModeMasterClient, [693](#)
- IsShutdown, [693](#)
- IsSimulationUpdating, [693](#)
- IsSinglePlayer, [693](#)
- IsStarting, [693](#)
- JoinSessionLobby, [658](#)
- LagCompensation, [693](#)
- LatestServerTick, [694](#)
- LoadScene, [659](#), [660](#)
- LobbyInfo, [694](#)
- LocalAlpha, [694](#)
- LocalPlayer, [694](#)
- LocalRenderTime, [694](#)
- MakeDontDestroyOnLoad, [660](#)
- Mode, [694](#)
- MoveGameObjectToSameScene, [660](#)
- MoveGameObjectToScene, [661](#)
- MoveToRunnerScene, [661](#)
- MoveToRunnerScene< T >, [662](#)
- NATType, [695](#)
- ObjectAcquired, [697](#)
- ObjectDelegate, [662](#)
- ObjectProvider, [695](#)
- OnBeforeSpawned, [662](#)
- Prefabs, [695](#)
- ProvideInput, [695](#)
- PushHostMigrationSnapshot, [662](#)
- RegisterSceneObjects, [662](#)
- Release, [641](#)
- ReleaseStateAuthority, [663](#)
- RemoteRenderTime, [695](#)
- RemoveCallbacks, [663](#)
- RemoveGlobal, [664](#)
- RenderInternal, [664](#)
- RequestStateAuthority, [664](#)
- Running, [641](#)
- SceneManager, [695](#)
- SendReliableDataToPlayer, [664](#)
- SendReliableDataToServer, [665](#)
- SendRpc, [665](#)
- SessionInfo, [696](#)
- SetAreaOfInterestCellSize, [665](#)
- SetAreaOfInterestGrid, [666](#)
- SetBehaviourReplicateTo, [666](#)
- SetBehaviourReplicateToAll, [667](#)
- SetIsSimulated, [667](#)
- SetMasterClient, [667](#)
- SetPlayerAlwaysInterested, [668](#)
- SetPlayerObject, [668](#)
- SetSimulateMultiPeerPhysics, [668](#)
- Shutdown, [641](#), [669](#)
- SimulationTime, [696](#)
- SimulationUnityScene, [696](#)
- SinglePlayerContinue, [669](#)
- SinglePlayerPause, [669](#)
- Spawn, [669](#)–[672](#)
- Spawn< T >, [672](#)
- SpawnAsync, [673](#)–[675](#)
- SpawnAsync< T >, [676](#)
- Stage, [696](#)
- StartGame, [677](#)
- Starting, [641](#)
- State, [696](#)
- States, [641](#)
- Tick, [696](#)
- TickRate, [697](#)
- TicksExecuted, [697](#)
- Topology, [697](#)
- TryFindBehaviour, [677](#)
- TryFindBehaviour< T >, [678](#)
- TryFindObject, [678](#)
- TryGetBehaviourStatistics, [679](#)
- TryGetFusionStatistics, [679](#)
- TryGetInputForPlayer< T >, [679](#)
- TryGetNetworkedBehaviourFromNetworkedObjectRef< T >, [680](#)
- TryGetNetworkedBehaviourId, [680](#)
- TryGetObjectRefFromNetworkedBehaviour, [681](#)
- TryGetPhysicsInfo, [681](#)
- TryGetPlayerObject, [681](#)
- TryGetSceneInfo, [682](#)
- TrySetPhysicsInfo, [682](#)
- TrySpawn, [683](#), [685](#), [686](#)
- TrySpawn< T >, [687](#)
- UnloadScene, [687](#)
- UpdateInternal, [689](#)
- UserId, [697](#)
- NetworkRunnerCallbackArgs, [698](#)
- NetworkRunnerCallbackArgs.ConnectRequest, [698](#)
 - Accept, [698](#)
 - Refuse, [699](#)
 - RemoteAddress, [699](#)
 - Waiting, [699](#)
- NetworkRunnerUpdaterDefault, [699](#)
 - RegisterInPlayerLoop, [700](#)
 - RenderSettings, [701](#)
 - UnregisterFromPlayerLoop, [700](#)
 - UpdateSettings, [701](#)
- NetworkRunnerUpdaterDefault.NetworkRunnerRender, [701](#)

- NetworkRunnerUpdaterDefault.NetworkRunnerUpdate, 702
- NetworkRunnerUpdaterDefault.InvokeSettings, 702
 - AddMode, 705
 - Equals, 703
 - GetHashCode, 703
 - operator!=, 703
 - operator==, 704
 - ReferencePlayerLoopSystem, 705
 - ToString, 704
- NetworkSceneAsyncOp, 705
 - AddOnCompleted, 706
 - Error, 709
 - FromAsyncOperation, 706
 - FromCompleted, 707
 - FromCoroutine, 707
 - FromError, 708
 - FromTask, 708
 - GetAwaiter, 708
 - IsDone, 709
 - IsValid, 709
 - SceneRef, 709
- NetworkSceneAsyncOp.Awaiter, 710
 - Awaiter, 710
 - GetResult, 711
 - IsCompleted, 711
 - OnCompleted, 711
- NetworkSceneInfo, 711
 - AddSceneRef, 713
 - Equals, 713
 - GetHashCode, 714
 - IndexOf, 714
 - MaxScenes, 715
 - operator NetworkSceneInfo, 714
 - RemoveSceneRef, 715
 - SceneCount, 716
 - SceneParams, 716
 - Scenes, 716
 - SIZE, 715
 - ToString, 715
 - Version, 716
 - WORD_COUNT, 716
- NetworkSceneInfoChangeSource
 - Fusion, 51
- NetworkSceneInfoDefaultFlags
 - Fusion, 51
- NetworkSceneLoadId, 717
 - Equals, 718
 - GetHashCode, 718
 - NetworkSceneLoadId, 717
 - operator NetworkSceneLoadId, 718
 - operator!=, 719
 - operator==, 719
 - ToString, 719
 - Value, 720
- NetworkSceneObjectId, 720
 - Equals, 721, 722
 - GetHashCode, 722
 - IsValid, 723
 - LoadId, 722
 - NetworkSceneObjectId, 721
 - ObjectId, 722
 - Scene, 723
 - SceneLoadId, 723
 - ToString, 722
- NetworkSerialize
 - NetworkBehaviour, 399
- NetworkSerializeMethodAttribute, 723
 - MaxSize, 724
- NetworkSimulationConfiguration, 724
 - AdditionalJitter, 725
 - AdditionalLoss, 726
 - Clone, 725
 - Create, 725
 - DelayMax, 726
 - DelayMin, 726
 - DelayPeriod, 726
 - DelayShape, 726
 - DelayThreshold, 726
 - Enabled, 727
 - LossChanceMax, 727
 - LossChanceMin, 727
 - LossChancePeriod, 727
 - LossChanceShape, 727
 - LossChanceThreshold, 727
- NetworkSpawnFlags
 - Fusion, 51
- NetworkSpawnOp, 728
 - GetAwaiter, 729
 - IsFailed, 729
 - IsQueued, 729
 - IsSpawned, 729
 - Object, 729
 - Runner, 729
 - Status, 730
- NetworkSpawnOp.Awaiter, 730
 - Awaiter, 730
 - GetResult, 731
 - IsCompleted, 731
 - OnCompleted, 731
- NetworkSpawnStatus
 - Fusion, 51
- NetworkString
 - NetworkString< TSize >, 735
- NetworkString< TSize >, 732
 - Assign, 735
 - Capacity, 757
 - Compare, 735, 736
 - Compare< TOtherSize >, 736, 737
 - Contains, 737, 738
 - Contains< TOtherSize >, 739
 - EndsWith, 740
 - EndsWith< TOtherSize >, 740
 - Equals, 741, 742
 - Equals< TOtherSize >, 742, 743
 - Get, 743

- GetCapacity< TSize >, 744
- GetCharCount, 744
- GetEnumerator, 744
- GetHashCode, 744
- IndexOf, 745–747
- IndexOf< TOtherSize >, 748–750
- Length, 757
- NetworkString, 735
- operator NetworkString< TSize >, 750
- operator string, 751
- operator !=, 751, 752
- operator ==, 752, 753
- Set, 753
- StartsWith, 754
- StartsWith< TOtherSize >, 754
- Substring, 755
- this[int index], 757
- ToLower, 756
- ToString, 756
- ToUpper, 756
- Value, 757
- NetworkStructUtils, 757
 - GetWordCount< T >, 758
- NetworkStructWeavedAttribute, 758
 - NetworkStructWeavedAttribute, 759
 - WordCount, 759
- NetworkTransform, 759
 - AutoUpdateAreaOfInterestOverride, 762
 - DisableSharedModelInterpolation, 761
 - SetAreaOfInterestOverride, 760
 - SyncParent, 761
 - SyncScale, 761
 - Teleport, 761
- NetworkTRSP, 762
 - Data, 765
 - IsMainTRSP, 765
 - reenabledTick, 764
 - Render, 763
 - ResolveAOIOVERRIDE, 763
 - SetAreaOfInterestOverride, 764
 - SetParentTransform, 764
 - State, 765
 - Teleport, 764
- NetworkTRSPData, 765
 - AreaOfInterestOverride, 766
 - NonNetworkedParent, 768
 - Parent, 766
 - Position, 767
 - POSITION_OFFSET, 767
 - Rotation, 767
 - Scale, 767
 - SIZE, 767
 - TeleportKey, 767
 - WORDS, 768
- NetworkTypeId
 - NetworkObject, 534
- NetworkTypeIdKind
 - Fusion, 52
- NetworkUnwrap
 - NetworkBehaviour, 399
 - NetworkObject, 531
- NetworkWrap
 - NetworkBehaviour, 400
 - NetworkObject, 531
- Next
 - NetConnectionMap.Iterator, 1038
 - NetworkRRNG, 625
 - ReliableHeader, 1059
 - Tick, 1094
- NextExclusive
 - NetworkRRNG, 625
- NextInt32
 - NetworkRRNG, 626
- NextPowerOfTwo
 - Maths, 329
- NextSingle
 - NetworkRRNG, 626
- NextSingleExclusive
 - NetworkRRNG, 626
- NextUInt32
 - NetworkRRNG, 626
- NoActiveConnections
 - Fusion, 56
- Noise
 - NetConfigSimulationOscillator, 1024
- NONE
 - AuthorityMasks, 110
- None
 - Fusion, 47, 49–51, 56, 60
 - Fusion.Analyzer, 63
 - Fusion.LagCompensation, 65
 - Fusion.Protocol, 66
 - NetConnectionMap, 1032
 - NetworkBehaviourId, 437
 - NetworkProjectConfig, 613
 - PlayerRef, 778
 - SceneRef, 887
 - TickTimer, 1120
- NonNetworkedParent
 - NetworkTRSPData, 768
- Normal
 - LagCompensatedHit, 279
 - LagCompensationUtils.ContactData, 293
- Normalized
 - Fusion, 60
- NormalizedPercentage
 - Fusion, 60
- NormalizedRectAttribute, 768
 - AspectRatio, 769
 - InvertY, 769
 - NormalizedRectAttribute, 769
- NoSimulation
 - Simulation, 914
- NotCulled
 - Fusion, 56
- NotEqual

- Fusion, [47](#)
- NotFound
 - Fusion, [50](#)
 - TickRate, [1106](#)
- Notify
 - NetConfig, [1017](#)
- NotifyLocalSimulationNotAllowedToSendRpc
 - NetworkBehaviourUtils, [447](#)
- NotifyLocalTargetedRpcCulled
 - NetworkBehaviourUtils, [448](#)
- NotifyNetworkUnwrapFailed< T >
 - NetworkBehaviourUtils, [448](#)
- NotifyNetworkWrapFailed< T >
 - NetworkBehaviourUtils, [448](#), [449](#)
- NotifyRpcPayloadSizeExceeded
 - NetworkBehaviourUtils, [449](#)
- NotifyRpcTargetUnreachable
 - NetworkBehaviourUtils, [449](#)
- NotInvokableDuringResim
 - Fusion, [54](#), [56](#)
- NotInvokableLocally
 - Fusion, [54](#)
- NotSentBroadcastNoActiveConnections
 - Fusion, [56](#)
- NotSentBroadcastNoConfirmedNorInterestedClients
 - Fusion, [56](#)
- NotSentTargetClientNotAvailable
 - Fusion, [56](#)
- NotSentTargetObjectNotConfirmed
 - Fusion, [56](#)
- NotSentTargetObjectNotInPlayerInterest
 - Fusion, [56](#)
- NotZero
 - Fusion, [47](#)
- Null
 - Ptr, [832](#)
- NullChecksForNetworkedProperties
 - NetworkProjectConfig, [618](#)
- Object
 - FusionGlobalScriptableObjectLoadResult, [184](#)
 - NetworkBehaviourId, [437](#)
 - NetworkObjectReleaseContext, [569](#)
 - NetworkSpawnOp, [729](#)
 - RpcHeader, [867](#)
 - SimulationBehaviour, [927](#)
- ObjectAcquired
 - NetworkRunner, [697](#)
- ObjectCount
 - Simulation, [919](#)
- ObjectDataConsistency
 - SimulationConfig, [932](#)
- ObjectDelegate
 - NetworkRunner, [662](#)
- ObjectId
 - NetworkSceneObjectId, [722](#)
- ObjectInitializer
 - StartGameArgs, [1073](#)
- ObjectProvider
 - NetworkRunner, [695](#)
 - StartGameArgs, [1073](#)
- Objects
 - Simulation, [919](#)
- ObjectsAllocMemoryFreeInBytes
 - FusionStatisticsSnapshot, [1082](#)
- ObjectsAllocMemoryUsedInBytes
 - FusionStatisticsSnapshot, [1083](#)
- ObjectStatisticsManager
 - FusionStatisticsManager, [1079](#)
- ObjectType
 - FusionGlobalScriptableObjectSourceAttribute, [186](#)
- Offset
 - ColliderDrawInfo, [289](#)
 - Hitbox, [192](#)
 - HitboxRoot, [212](#)
 - NetBitBuffer.Offset, [999](#)
 - SimulationMessage, [952](#)
- offset
 - NetworkBehaviour, [403](#)
- OffsetBits
 - INetBitWriteStream, [958](#)
 - NetBitBuffer, [998](#)
 - NetBitBufferNull, [1006](#)
- OffsetBitsUnsafe
 - NetBitBuffer, [998](#)
- OffsetBytes
 - NetBitBuffer, [998](#)
- Ok
 - Fusion, [58](#)
 - Fusion.Sockets, [70](#)
 - StartGameResult, [1076](#)
 - TickRate, [1106](#)
- OnBeforeSpawned
 - NetworkRunner, [662](#)
- OnChangedRenderAttribute, [769](#)
 - MethodName, [770](#)
 - OnChangedRenderAttribute, [770](#)
- OnCompleted
 - NetworkSceneAsyncOp.Awaiter, [711](#)
 - NetworkSpawnOp.Awaiter, [731](#)
- OnConnected
 - INetPeerGroupCallbacks, [959](#)
- OnConnectedToServer
 - INetworkRunnerCallbacks, [243](#)
- OnConnectFailed
 - INetworkRunnerCallbacks, [243](#)
- OnConnectionAttempt
 - INetPeerGroupCallbacks, [959](#)
- OnConnectionFailed
 - INetPeerGroupCallbacks, [959](#)
- OnConnectionRequest
 - INetPeerGroupCallbacks, [960](#)
- OnConnectionRequestReply
 - Fusion.Sockets, [69](#)
- OnConnectRequest
 - INetworkRunnerCallbacks, [243](#)
- OnCustomAuthenticationResponse

- INetworkRunnerCallbacks, [243](#)
- OnDestroy
 - NetworkObject, [532](#)
- OnDisable
 - FusionGlobalScriptableObject< T >, [180](#)
 - NetworkProjectConfigAsset, [621](#)
- OnDisconnected
 - INetPeerGroupCallbacks, [960](#)
- OnDisconnectedFromServer
 - INetworkRunnerCallbacks, [244](#)
- OnDrawGizmos
 - Hitbox, [191](#)
 - HitboxRoot, [210](#)
- OnGameStarted
 - StartGameArgs, [1073](#)
- OnHostMigration
 - INetworkRunnerCallbacks, [244](#)
- OnInput
 - INetworkRunnerCallbacks, [244](#)
- OnInputMissing
 - INetworkRunnerCallbacks, [244](#)
- OnLoadedAsGlobal
 - FusionGlobalScriptableObject< T >, [180](#)
- OnNotifyData
 - INetPeerGroupCallbacks, [960](#)
- OnNotifyDelivered
 - INetPeerGroupCallbacks, [962](#)
- OnNotifyDispose
 - INetPeerGroupCallbacks, [962](#)
- OnNotifyLost
 - INetPeerGroupCallbacks, [962](#)
- OnObjectEnterAOI
 - INetworkRunnerCallbacks, [245](#)
- OnObjectExitAOI
 - INetworkRunnerCallbacks, [245](#)
- OnPlayerJoined
 - INetworkRunnerCallbacks, [245](#)
- OnPlayerLeft
 - INetworkRunnerCallbacks, [245](#)
- OnReliableData
 - INetPeerGroupCallbacks, [963](#)
- OnReliableDataProgress
 - INetworkRunnerCallbacks, [246](#)
- OnReliableDataReceived
 - INetworkRunnerCallbacks, [246](#)
- OnSceneInfoChanged
 - INetworkSceneManager, [251](#)
- OnSceneLoadDone
 - INetworkRunnerCallbacks, [246](#)
- OnSceneLoadStart
 - INetworkRunnerCallbacks, [247](#)
- OnSessionListUpdated
 - INetworkRunnerCallbacks, [247](#)
- OnShutdown
 - INetworkRunnerCallbacks, [247](#)
- OnUnconnectedData
 - INetPeerGroupCallbacks, [963](#)
- OnUnloadedAsGlobal
 - FusionGlobalScriptableObject< T >, [180](#)
- OnUnreliableData
 - INetPeerGroupCallbacks, [963](#)
- OnUserSimulationMessage
 - INetworkRunnerCallbacks, [248](#)
- OpenInternet
 - Fusion.Sockets.Stun, [70](#)
- OperationCanceled
 - Fusion, [59](#)
- OperationExpireTime
 - NetConfig, [1017](#)
- OperationTimeout
 - Fusion, [59](#)
- operator Angle
 - Angle, [93](#), [94](#)
- operator bool
 - NetworkBehaviourBuffer, [416](#)
 - NetworkBehaviourBufferInterpolator, [428](#)
 - NetworkBool, [458](#)
 - NetworkId, [497](#)
 - Ptr, [830](#)
 - SessionInfo, [902](#)
 - Tick, [1095](#)
- operator double
 - Angle, [94](#)
- operator float
 - Angle, [96](#)
 - FloatCompressed, [175](#)
- operator FloatCompressed
 - FloatCompressed, [176](#)
- operator FusionGlobalScriptableObjectLoadResult
 - FusionGlobalScriptableObjectLoadResult, [184](#)
- operator Guid
 - NetworkObjectGuid, [545](#)
 - NetworkPrefabRef, [597](#)
- operator int
 - Tick, [1095](#)
- operator LagCompensatedHit
 - LagCompensatedHit, [277](#)
- operator long
 - Mask256, [315](#)
- operator Mask256
 - FieldsMask< T >, [162](#)
 - Mask256, [315](#)
- operator NetworkArray< T >
 - NetworkBehaviourUtils.ArrayInitializer< T >, [453](#)
- operator NetworkArrayReadOnly< T >
 - NetworkArray< T >, [377](#)
- operator NetworkBehaviourId
 - NetworkBehaviour, [400](#)
- operator NetworkBool
 - NetworkBool, [458](#)
- operator NetworkDictionary< K, V >
 - NetworkBehaviourUtils.DictionaryInitializer< K, V >, [454](#)
- operator NetworkDictionaryReadOnly< K, V >
 - NetworkDictionary< K, V >, [477](#)
- operator NetworkId

- NetworkObject, [532](#)
- operator NetworkLinkedList< T >
 - NetworkBehaviourUtils.ArrayInitializer< T >, [453](#)
- operator NetworkObjectGuid
 - NetworkObjectGuid, [546](#)
 - NetworkPrefabRef, [597](#)
- operator NetworkObjectHeaderPtr
 - NetworkObjectHeaderPtr, [558](#)
- operator NetworkObjectTypeId
 - NetworkObjectTypeId, [576](#)
- operator NetworkPrefabRef
 - NetworkObjectGuid, [546](#)
 - NetworkPrefabRef, [597](#)
- operator NetworkSceneInfo
 - NetworkSceneInfo, [714](#)
- operator NetworkSceneLoadId
 - NetworkSceneLoadId, [718](#)
- operator NetworkString< TSize >
 - NetworkString< TSize >, [750](#)
- operator Quaternion
 - QuaternionCompressed, [836](#)
- operator QuaternionCompressed
 - QuaternionCompressed, [836](#)
- operator SerializableType
 - SerializableType< BaseType >, [897](#)
- operator string
 - NetworkString< TSize >, [751](#)
- operator Tick
 - Tick, [1095](#)
- operator Type
 - SerializableType< BaseType >, [898](#)
- operator Vector2
 - Vector2Compressed, [1142](#)
 - Vector3Compressed, [1147](#)
- operator Vector2Compressed
 - Vector2Compressed, [1143](#)
- operator Vector3
 - Vector3Compressed, [1147](#)
- operator Vector3Compressed
 - Vector3Compressed, [1148](#)
- operator Vector4
 - Vector4Compressed, [1152](#)
- operator Vector4Compressed
 - Vector4Compressed, [1153](#)
- operator!=
 - Angle, [96](#)
 - FloatCompressed, [176](#)
 - NetConnectionId, [1029](#)
 - NetworkBehaviourId, [436](#)
 - NetworkId, [498](#)
 - NetworkLoadSceneParameters, [521](#)
 - NetworkObjectGuid, [547](#)
 - NetworkObjectHeader, [554](#)
 - NetworkObjectTypeId, [577](#)
 - NetworkPrefabId, [588](#)
 - NetworkPrefabRef, [598](#)
 - NetworkRunnerUpdaterDefaultInvokeSettings, [703](#)
 - NetworkSceneLoadId, [719](#)
 - NetworkString< TSize >, [751](#), [752](#)
 - PlayerRef, [775](#)
 - Ptr, [831](#)
 - QuaternionCompressed, [837](#)
 - SceneRef, [884](#)
 - Tick, [1097](#)
 - Vector2Compressed, [1143](#)
 - Vector3Compressed, [1149](#)
 - Vector4Compressed, [1153](#)
- NetworkString< TSize >, [751](#), [752](#)
- PlayerRef, [775](#)
- Ptr, [830](#)
- QuaternionCompressed, [837](#)
- SceneRef, [884](#)
- Tick, [1097](#)
- Vector2Compressed, [1143](#)
- Vector3Compressed, [1148](#)
- Vector4Compressed, [1153](#)
- operator<
 - Angle, [97](#)
 - Tick, [1097](#)
- operator<=
 - Angle, [98](#)
 - Tick, [1097](#)
- operator>
 - Angle, [98](#)
 - Tick, [1097](#)
- operator>=
 - Angle, [99](#)
 - Tick, [1098](#)
- operator~
 - Mask256, [316](#)
- operator+
 - Angle, [96](#)
 - Ptr, [831](#)
- operator-
 - Angle, [97](#)
 - Ptr, [831](#)
- operator==
 - Angle, [98](#)
 - FloatCompressed, [177](#)
 - NetConnectionId, [1029](#)
 - NetworkBehaviourId, [436](#)
 - NetworkId, [498](#)
 - NetworkLoadSceneParameters, [522](#)
 - NetworkObjectGuid, [547](#)
 - NetworkObjectHeader, [554](#)
 - NetworkObjectTypeId, [577](#)
 - NetworkPrefabId, [588](#)
 - NetworkPrefabRef, [598](#)
 - NetworkRunnerUpdaterDefaultInvokeSettings, [704](#)
 - NetworkSceneLoadId, [719](#)
 - NetworkString< TSize >, [752](#), [753](#)
 - PlayerRef, [775](#)
 - Ptr, [831](#)
 - QuaternionCompressed, [837](#)
 - SceneRef, [884](#)
 - Tick, [1097](#)
 - Vector2Compressed, [1143](#)
 - Vector3Compressed, [1149](#)
 - Vector4Compressed, [1153](#)
- operator&
 - Mask256, [315](#)
- operator |
 - Mask256, [315](#)
- Optimize
 - LagCompensationSettings, [311](#)

- Options
 - NetworkPrefabTable, [609](#)
 - Query, [296](#)
 - QueryParams, [298](#)
- Order
 - FusionGlobalScriptableObjectSourceAttribute, [186](#)
- order
 - UnityContextMenuAttribute, [1127](#)
 - UnityDelayedAttribute, [1127](#)
 - UnityHeaderAttribute, [1129](#)
 - UnityMinAttribute, [1130](#)
 - UnityMultilineAttribute, [1130](#)
 - UnityNonReorderableAttribute, [1131](#)
 - UnityRangeAttribute, [1132](#)
 - UnitySpaceAttribute, [1135](#)
 - UnityTooltipAttribute, [1136](#)
- Origin
 - RaycastQuery, [303](#)
 - RaycastQueryParams, [304](#)
- OutBandwidth
 - FusionStatisticsSnapshot, [1083](#)
 - NetworkObjectStatisticsSnapshot, [1091](#)
- OutObjectUpdates
 - FusionStatisticsSnapshot, [1083](#)
- OutPackets
 - FusionStatisticsSnapshot, [1083](#)
 - NetworkObjectStatisticsSnapshot, [1091](#)
- Overflow
 - NetBitBuffer, [998](#)
- Overflowing
 - BitStream, [823](#)
- OverlapBox
 - HitboxManager, [196–198](#)
- OverlapSphere
 - HitboxManager, [198–200](#)
- Packets
 - Fusion, [60](#)
- PacketSize
 - NetConfig, [1018](#)
- PacketSizeInBits
 - NetConfig, [1019](#)
- PadToByteBoundary
 - NetBitBuffer, [980](#)
 - NetBitBufferNull, [1003](#)
- PadToByteBoundaryAndGetPtr
 - NetBitBuffer, [981](#)
- PageCount
 - HeapConfiguration, [188](#)
- PageShift
 - HeapConfiguration, [188](#)
- PageSizes
 - Fusion, [52](#)
- Parent
 - NetworkTRSPData, [766](#)
- Parse
 - NetworkObjectGuid, [547](#)
 - NetworkPrefabRef, [598](#)
 - SceneRef, [884](#)
- PayloadSizeExceeded
 - Fusion, [54, 56](#)
- Peek
 - NetworkRNG, [629](#)
- PeekBoolean
 - NetBitBuffer, [981](#)
- PeerMode
 - NetworkProjectConfig, [618](#)
- PeerModes
 - NetworkProjectConfig, [613](#)
- Pending
 - TickAccumulator, [1104](#)
- Penetration
 - LagCompensationUtils.ContactData, [293](#)
- Percentage
 - Fusion, [60](#)
- Period
 - NetConfigSimulationOscillator, [1025](#)
- PerSecond
 - Fusion, [60](#)
- Photon
 - Fusion, [58](#)
- PhotonCloudTimeout
 - Fusion, [59](#)
- PhysX
 - Fusion.LagCompensation, [65](#)
- Player
 - Fusion, [53](#)
 - Query, [296](#)
 - QueryParams, [299](#)
 - SimulationInput, [935](#)
- PLAYER_DATA_WORD
 - NetworkObjectHeader, [556](#)
- PlayerCount
 - SessionInfo, [904](#)
 - SimulationConfig, [932](#)
 - StartGameArgs, [1074](#)
- PlayerData
 - NetworkObjectTypeId, [581](#)
- PlayerId
 - PlayerRef, [779](#)
- PlayerJoined
 - IPlayerJoined, [270](#)
- PlayerLeft
 - IPlayerLeft, [271](#)
- PlayerMaxCount
 - SimulationRuntimeConfig, [954](#)
- PlayerRef, [771](#)
 - AsIndex, [777](#)
 - Comparer, [777](#)
 - Equals, [772, 773](#)
 - FromEncoded, [773](#)
 - FromIndex, [773](#)
 - GetHashCode, [775](#)
 - Invalid, [778](#)
 - IsMasterClient, [778](#)
 - IsNone, [778](#)
 - IsRealPlayer, [778](#)

- MASTER_CLIENT_RAW, [777](#)
- MasterClient, [778](#)
- None, [778](#)
- operator!=, [775](#)
- operator==, [775](#)
- PlayerId, [779](#)
- RawEncoded, [779](#)
- Read, [776](#)
- SIZE, [777](#)
- ToString, [776](#)
- Write, [776](#)
- Write< T >, [776](#)
- PlayMode
 - Fusion, [48](#)
- Point
 - LagCompensatedHit, [279](#)
 - LagCompensationUtils.ContactData, [293](#)
- Poll
 - ICommunicator, [825](#)
 - INetSocket, [966](#)
- PosBlendEnd
 - InterpolatedErrorCorrectionSettings, [268](#)
- PosBlendStart
 - InterpolatedErrorCorrectionSettings, [268](#)
- Position
 - BitStream, [823](#)
 - Hitbox, [193](#)
 - NetworkTRSPData, [767](#)
- POSITION_OFFSET
 - NetworkTRSPData, [767](#)
- PositionRotation
 - HitboxManager, [201](#)
- PositionRotationQueryParams, [293](#)
 - Hitbox, [294](#)
 - PositionRotationQueryParams, [294](#)
 - QueryParams, [294](#)
- PosMinCorrection
 - InterpolatedErrorCorrectionSettings, [269](#)
- PosTeleportDistance
 - InterpolatedErrorCorrectionSettings, [269](#)
- Prefab
 - Fusion, [52](#)
 - NetworkPrefabInfo, [592](#)
- PrefabId
 - NetworkPrefabAcquireContext, [584](#)
- PrefabOptions
 - NetworkProjectConfigAsset, [622](#)
- Prefabs
 - NetworkPrefabTable, [609](#)
 - NetworkProjectConfigAsset, [622](#)
 - NetworkRunner, [695](#)
- PrefabTable
 - NetworkProjectConfig, [618](#)
- PreProcessingDelegate
 - Fusion.LagCompensation, [65](#)
 - Query, [297](#)
 - QueryParams, [299](#)
- PreserveInPluginAttribute, [779](#)
 - KeepNonStateMembers, [780](#)
 - PreserveInPluginAttribute, [779](#)
- Prev
 - ReliableHeader, [1059](#)
- Primes, [780](#)
 - GetNextPrime, [780](#), [781](#)
 - IsPrime, [781](#)
- PrintBits
 - Maths, [329](#)
- Priority
 - EditorButtonAttribute, [141](#)
- PriorityCallback
 - NetworkObject, [535](#)
- PriorityLevel
 - Fusion, [52](#)
- PriorityLevelDelegate
 - NetworkObject, [532](#)
- ProjectConfig
 - Simulation, [919](#)
- Properties
 - SessionInfo, [904](#)
- PropertyAttribute, [782](#)
- PropertyName
 - DefaultForPropertyAttribute, [123](#)
- PropertyReader
 - NetworkBehaviour.PropertyReader< T >, [414](#)
- ProvideInput
 - NetworkRunner, [695](#)
- Proxies
 - Fusion, [57](#)
- PROXY
 - AuthorityMasks, [110](#)
- Ptr, [828](#)
 - Address, [832](#)
 - Equals, [829](#)
 - GetHashCode, [830](#)
 - NetworkObjectHeaderPtr, [558](#)
 - Null, [832](#)
 - operator bool, [830](#)
 - operator!=, [830](#)
 - operator+, [831](#)
 - operator-, [831](#)
 - operator==, [831](#)
 - SIZE, [832](#)
 - ToString, [832](#)
- Ptr.EqualityComparer, [833](#)
 - Equals, [833](#)
 - GetHashCode, [833](#)
- PushHostMigrationSnapshot
 - NetworkRunner, [662](#)
- PushPackage
 - ICommunicator, [825](#)
- Quaternion
 - NetworkBehaviourBufferInterpolator, [429](#)
- QuaternionCompress
 - Maths, [329](#)
- QuaternionCompressed, [834](#)
 - Equals, [835](#)

- GetHashCode, [836](#)
- operator Quaternion, [836](#)
- operator QuaternionCompressed, [836](#)
- operator !=, [837](#)
- operator ==, [837](#)
- W, [838](#)
- wEncoded, [838](#)
- X, [838](#)
- xEncoded, [838](#)
- Y, [839](#)
- yEncoded, [838](#)
- Z, [839](#)
- zEncoded, [838](#)
- QuaternionDecompress
 - Maths, [330](#)
- Query, [294](#)
 - Alpha, [296](#)
 - Check, [296](#)
 - LayerMask, [296](#)
 - Options, [296](#)
 - Player, [296](#)
 - PreProcessingDelegate, [297](#)
 - Query, [295](#)
 - Tick, [297](#)
 - TickTo, [297](#)
 - TriggerInteraction, [297](#)
 - UserArgs, [297](#)
- QueryParams, [298](#)
 - Alpha, [298](#)
 - BoxOverlapQueryParams, [285](#)
 - LayerMask, [298](#)
 - Options, [298](#)
 - Player, [299](#)
 - PositionRotationQueryParams, [294](#)
 - PreProcessingDelegate, [299](#)
 - RaycastQueryParams, [304](#)
 - SphereOverlapQueryParams, [309](#)
 - Tick, [299](#)
 - TickTo, [299](#)
 - TriggerInteraction, [299](#)
 - UserArgs, [299](#)
- Queued
 - Fusion, [52](#)
- Radians
 - Fusion, [60](#)
- RadiansPerSecond
 - Fusion, [60](#)
- Radius
 - ColliderDrawInfo, [289](#)
 - SphereOverlapQuery, [307](#)
 - SphereOverlapQueryParams, [309](#)
- RangeExAttribute, [839](#)
 - ClampMax, [840](#)
 - ClampMin, [840](#)
 - Max, [841](#)
 - Min, [841](#)
 - RangeExAttribute, [840](#)
 - UseSlider, [840](#)
- RangeExclusive
 - NetworkRNG, [627](#)
- RangeInclusive
 - NetworkRNG, [627](#), [628](#)
- Raw
 - NetworkId, [500](#)
 - Tick, [1098](#)
- RawEncoded
 - PlayerRef, [779](#)
- RawGuidValue
 - NetworkObjectGuid, [549](#)
 - NetworkPrefabRef, [600](#)
- RawValue
 - NetworkPrefabId, [589](#)
 - SceneRef, [885](#)
- Raycast
 - HitboxManager, [202](#), [203](#)
- RaycastAll
 - HitboxManager, [204–206](#)
- RaycastAllQuery, [300](#)
 - RaycastAllQuery, [300](#)
- RaycastQuery, [301](#)
 - Check, [302](#)
 - Direction, [302](#)
 - Length, [302](#)
 - Origin, [303](#)
 - RaycastQuery, [302](#)
- RaycastQueryParams, [303](#)
 - Direction, [304](#)
 - Length, [304](#)
 - Origin, [304](#)
 - QueryParams, [304](#)
 - RaycastQueryParams, [303](#)
 - StaticHitsCapacity, [304](#)
- Read
 - IElementReaderWriter< T >, [229](#)
 - IUnitySurrogate, [255](#)
 - NetworkBehaviour.ArrayReader< T >, [405](#)
 - NetworkBehaviour.BehaviourReader< T >, [406](#)
 - NetworkBehaviour.DictionaryReader< K, V >, [412](#)
 - NetworkBehaviour.LinkListReader< T >, [413](#)
 - NetworkBehaviour.PropertyReader< T >, [415](#)
 - NetworkBehaviourBuffer, [417](#), [419](#)
 - NetworkId, [498](#)
 - PlayerRef, [776](#)
 - RpcHeader, [865](#)
 - UnityArraySurrogate< T, ReaderWriter >, [258](#)
 - UnityDictionarySurrogate< TKeyType, TKeyReaderWriter, TValueType, TValueReaderWriter >, [260](#)
 - UnityLinkedListSurrogate< T, ReaderWriter >, [262](#)
 - UnitySurrogateBase, [264](#)
 - UnityValueSurrogate< T, TReaderWriter >, [266](#)
- Read< T >
 - NetworkBehaviourBuffer, [419](#), [421](#)
- ReadBool
 - BitStream, [790](#)
- ReadBoolean

- BitStream, [791](#)
- NetBitBuffer, [981](#)
- ReadWriteUtilsForWeaver, [849](#)
- ReadByte
 - BitStream, [791](#)
 - NetBitBuffer, [981](#)
- ReadByteArray
 - BitStream, [792](#), [793](#)
- ReadByteArrayLengthPrefixed
 - BitStream, [793](#)
- ReadBytesAligned
 - NetBitBuffer, [982](#)
- ReadChar
 - BitStream, [793](#)
- ReadDouble
 - BitStream, [793](#)
 - NetBitBuffer, [982](#)
- Reader
 - NetBitBufferSerializer, [1008](#)
- ReadFloat
 - BitStream, [794](#)
 - ReadWriteUtils, [843](#)
- ReadGuid
 - BitStream, [794](#)
- Reading
 - BitStream, [824](#)
 - NetBitBufferSerializer, [1011](#)
- ReadInt
 - BitStream, [794](#)
 - SimulationMessage, [946](#)
- ReadInt16
 - NetBitBuffer, [982](#)
- ReadInt32
 - NetBitBuffer, [983](#)
- ReadInt32VarLength
 - NetBitBuffer, [983](#)
- ReadInt64
 - NetBitBuffer, [984](#)
- ReadInt64VarLength
 - NetBitBuffer, [984](#)
- ReadInt_Shifted
 - BitStream, [795](#)
- ReadLengthPrefixedUTF8
 - Native, [360](#)
- ReadLong
 - BitStream, [795](#)
- ReadNetworkedObjectRef
 - SimulationMessage, [946](#)
- ReadOnly
 - Fusion, [48](#)
- ReadOnlyAttribute, [841](#)
 - InEditMode, [841](#)
 - InPlayMode, [842](#)
- ReadQuaternion
 - ReadWriteUtils, [843](#)
- ReadRef
 - IElementReaderWriter< T >, [229](#)
- ReadSByte
 - BitStream, [796](#)
- ReadShort
 - BitStream, [796](#)
- ReadSingle
 - NetBitBuffer, [984](#)
- ReadSize
 - RpcHeader, [865](#)
- ReadString
 - BitStream, [797](#)
 - NetBitBuffer, [985](#)
- ReadStringUtf32NoHash
 - ReadWriteUtilsForWeaver, [849](#)
- ReadStringUtf32WithHash
 - ReadWriteUtilsForWeaver, [849](#)
- ReadStringUtf8NoHash
 - ReadWriteUtilsForWeaver, [851](#)
- ReadUInt
 - BitStream, [797](#)
- ReadUInt16
 - NetBitBuffer, [985](#)
- ReadUInt32
 - NetBitBuffer, [986](#)
- ReadUInt32VarLength
 - NetBitBuffer, [986](#)
- ReadUInt64
 - NetBitBuffer, [987](#)
- ReadUInt64VarLength
 - NetBitBuffer, [987](#)
- ReadULong
 - BitStream, [798](#)
- ReadUShort
 - BitStream, [798](#)
- ReadVector2
 - ReadWriteUtils, [843](#)
- ReadVector3
 - ReadWriteUtils, [844](#)
 - SimulationMessage, [946](#)
- ReadVector4
 - ReadWriteUtils, [844](#)
- ReadWriteUtils, [842](#)
 - ACCURACY, [846](#)
 - ReadFloat, [843](#)
 - ReadQuaternion, [843](#)
 - ReadVector2, [843](#)
 - ReadVector3, [844](#)
 - ReadVector4, [844](#)
 - WriteFloat, [844](#)
 - WriteQuaternion, [845](#)
 - WriteVector2, [845](#)
 - WriteVector3, [845](#)
 - WriteVector4, [846](#)
- ReadWriteUtilsForWeaver, [846](#)
 - GetByteArrayHashCode, [847](#)
 - GetByteCountUtf8NoHash, [847](#)
 - GetStringHashCode, [848](#)
 - GetWordCountString, [848](#)
 - ReadBoolean, [849](#)
 - ReadStringUtf32NoHash, [849](#)

- ReadStringUtf32WithHash, [849](#)
- ReadStringUtf8NoHash, [851](#)
- VerifyRawNetworkUnwrap< T >, [851](#)
- VerifyRawNetworkWrap< T >, [852](#)
- WriteBoolean, [852](#)
- WriteStringUtf32NoHash, [853](#)
- WriteStringUtf32WithHash, [853](#)
- WriteStringUtf8NoHash, [853](#)
- Receive
 - INetSocket, [967](#)
- ReceivePackage
 - ICommunicator, [825](#)
- Recv
 - NetPeer, [1042](#)
- Redundancy
 - SimulationConfig, [931](#)
- RedundancyUncompressed
 - SimulationConfig, [931](#)
- RedundantInputs
 - TimeSyncConfiguration, [1123](#)
- RedundantSnapshots
 - TimeSyncConfiguration, [1124](#)
- reenabledTick
 - NetworkTRSP, [764](#)
- ReferenceCountAdd
 - SimulationMessage, [947](#)
- ReferenceCountSub
 - SimulationMessage, [947](#)
- ReferencePlayerLoopSystem
 - NetworkRunnerUpdaterDefaultInvokeSettings, [705](#)
- References
 - SimulationMessage, [952](#)
- ReferenceToPointer< T >
 - Native, [360](#)
- RefitBVHTime
 - LagCompensationStatisticsSnapshot, [1087](#)
- ReflectionUtils, [854](#)
 - GetAllNetworkBehaviourTypes, [854](#)
 - GetAllSimulationBehaviourTypes, [855](#)
 - GetAllWeavedAssemblies, [855](#)
 - GetAllWeavedNetworkBehaviourTypes, [855](#)
 - GetAllWeavedSimulationBehaviourTypes, [855](#)
 - GetAllWeaverGeneratedTypes, [856](#)
 - GetCustomAttributeOrThrow< T >, [856](#)
 - GetWeavedAttributeOrThrow, [857](#)
- Refuse
 - Fusion.Sockets, [70](#)
 - NetworkRunnerCallbackArgs.ConnectRequest, [699](#)
- Region
 - LobbyInfo, [312](#)
 - SessionInfo, [904](#)
- RegisterInPlayerLoop
 - NetworkRunnerUpdaterDefault, [700](#)
- RegisterMetaData
 - NetworkBehaviourUtils, [450](#)
- RegisterPackageCallback< T >
 - ICommunicator, [826](#)
- RegisterRpcInvokeDelegates
 - NetworkBehaviourUtils, [450](#)
- RegisterSceneObjects
 - NetworkRunner, [662](#)
- ReinitializeHitboxesBeforeRegistration
 - HitboxRoot, [209](#)
- ReinterpretState< T >
 - NetworkBehaviour, [401](#)
 - NetworkBehaviourBuffer, [421](#)
- Relayed
 - Fusion, [47](#)
- Release
 - INetworkAssetSource< T >, [235](#)
 - NetBitBuffer, [987](#)
 - NetworkRunner, [641](#)
- ReleaseInstance
 - INetworkObjectProvider, [240](#)
- ReleaseRef
 - NetBitBuffer, [988](#)
- ReleaseStateAuthority
 - NetworkObject, [532](#)
 - NetworkRunner, [663](#)
- Reliable
 - Fusion, [54](#)
- ReliableDataTransferModes
 - NetworkConfiguration, [470](#)
- ReliableDataTransfers
 - NetworkConfiguration, [469](#)
- ReliableHeader, [1058](#)
 - GetData, [1058](#)
 - Id, [1059](#)
 - Next, [1059](#)
 - Prev, [1059](#)
 - SIZE, [1059](#)
- ReliableId, [1059](#)
 - _padding, [1060](#)
 - Key, [1060](#)
 - Sequence, [1061](#)
 - SIZE, [1061](#)
 - SliceLength, [1061](#)
 - Source, [1061](#)
 - SourceCombined, [1062](#)
 - SourceSend, [1061](#)
 - Target, [1061](#)
 - TotalLength, [1062](#)
- ReliableKey, [1062](#)
 - Data, [1064](#)
 - FromInts, [1063](#)
 - FromULongs, [1063](#)
 - GetInts, [1064](#)
 - GetULongs, [1064](#)
 - SIZE, [1064](#)
- ReliableList, [1065](#)
 - AddAfter, [1065](#)
 - AddBefore, [1066](#)
 - AddFirst, [1066](#)
 - AddLast, [1066](#)
 - Count, [1067](#)

- Head, [1067](#)
- Remove, [1066](#)
- RemoveHead, [1067](#)
- Tail, [1067](#)
- Remainder
 - TickAccumulator, [1104](#)
- RemainingTicks
 - TickTimer, [1118](#)
- RemainingTime
 - TickTimer, [1119](#)
- Remap
 - NetConnectionMap, [1035](#)
 - NetworkLinkedList< T >, [512](#)
- RemapAddress
 - NetPeer, [1043](#)
- Remote
 - Fusion, [51](#), [53](#), [57](#)
- RemoteAddress
 - NetConnection, [1027](#)
 - NetworkRunnerCallbackArgs.ConnectRequest, [699](#)
- RemoteAlpha
 - Simulation, [919](#)
- RemoteConnectionId
 - NetConnection, [1027](#)
- RemotePrefabCreated
 - IRemotePrefabCreated, [272](#)
- RemoteRenderTime
 - NetworkRunner, [695](#)
- RemoteTick
 - Simulation, [919](#)
- RemoteTickPrevious
 - Simulation, [920](#)
- Remove
 - NetBitBufferList, [1001](#)
 - NetConnectionMap, [1035](#)
 - NetworkDictionary< K, V >, [477](#), [478](#)
 - NetworkLinkedList< T >, [512](#)
 - ReliableList, [1066](#)
 - SerializableDictionary< TKey, TValue >, [892](#)
 - SimulationInput.Buffer, [939](#)
- RemoveCallbacks
 - NetworkRunner, [663](#)
- RemoveGlobal
 - NetworkRunner, [664](#)
- RemoveHead
 - NetBitBufferList, [1002](#)
 - ReliableList, [1067](#)
- RemoveInputAuthority
 - NetworkObject, [533](#)
- RemoveInstance
 - NetworkPrefabTable, [607](#)
- RemoveSceneRef
 - NetworkSceneInfo, [715](#)
- Render
 - NetworkTRSP, [763](#)
 - SimulationBehaviour, [926](#)
- RenderAttribute, [857](#)
- Method, [858](#)
- RenderAttribute, [858](#)
- Source, [858](#)
- Timeframe, [858](#)
- RenderExecutionCount
 - BehaviourStatisticsSnapshot, [1078](#)
- RenderExecutionTime
 - BehaviourStatisticsSnapshot, [1078](#)
- RenderInternal
 - NetworkRunner, [664](#)
- RenderSettings
 - NetworkRunnerUpdaterDefault, [701](#)
- RenderSource
 - Fusion, [53](#)
 - NetworkObject, [537](#)
- RenderTime
 - NetworkObject, [537](#)
- RenderTimeframe
 - Fusion, [53](#)
 - NetworkObject, [538](#)
- RenderTimeline, [859](#)
 - GetRenderBuffers, [859](#)
- RenderWeavedAttribute, [860](#)
 - RenderWeavedAttribute, [860](#)
- ReplaceDataFromBlockWithTemp
 - NetBitBuffer, [988](#)
- REPLICATE_WORD_ALIGN
 - Allocator, [82](#)
- REPLICATE_WORD_SHIFT
 - Allocator, [83](#)
- REPLICATE_WORD_SIZE
 - Allocator, [83](#)
- ReplicateTo
 - NetworkBehaviour, [401](#), [402](#)
 - NetworkObject, [535](#)
- ReplicateToAll
 - NetworkBehaviour, [402](#)
- ReplicateToDelegate
 - NetworkObject, [533](#)
- ReplicationFeatures
 - NetworkProjectConfig, [613](#)
 - SimulationConfig, [932](#)
- Requested
 - Fusion.Sockets, [69](#)
- RequestStateAuthority
 - NetworkObject, [533](#)
 - NetworkRunner, [664](#)
- RequiresUnsafeCode
 - AssemblyNameAttribute, [102](#)
- Reset
 - BitStream, [799](#)
 - FixedArray< T >.Enumerator, [171](#)
 - NetworkArray< T >.Enumerator, [380](#)
 - NetworkBehaviour.ChangeDetector.Enumerator, [411](#)
 - NetworkDictionary< K, V >.Enumerator, [481](#)
 - NetworkLinkedList< T >.Enumerator, [515](#)
 - RuntimeUnityFlagsSetup, [877](#)

- SerializableDictionary< TKey, TValue >, [892](#)
- StaticFieldAttribute, [89](#)
- Timer, [1121](#)
- UTF32Tools.CharEnumerator, [1139](#)
- ResetFast
 - BitStream, [799](#)
- ResetMethod
 - Fusion.Analyzer, [63](#)
- ResetState
 - NetworkBehaviour, [402](#)
- Resimulate
 - Fusion, [60](#)
- Resimulations
 - EngineProfiler, [151](#)
 - FusionStatisticsSnapshot, [1083](#)
- ResimulationsCallback
 - EngineProfiler, [155](#)
- Resolve
 - TickRate, [1109](#)
- ResolveAOIOVERRIDE
 - NetworkTRSP, [763](#)
- ResolveNetworkPrefabSourceAttribute, [860](#)
- ResourceType
 - UnityResourcePathAttribute, [1133](#)
- Restart
 - Timer, [1121](#)
- Result
 - RpcSendResult, [874](#)
- Retry
 - Fusion, [49](#)
- ReverseSaw
 - NetConfigSimulationOscillator, [1024](#)
- Root
 - Hitbox, [192](#)
- RootGameObjects
 - SceneLoadDoneArgs, [878](#)
- Rotation
 - BoxOverlapQuery, [284](#)
 - BoxOverlapQueryParams, [285](#)
 - NetworkTRSPData, [767](#)
- RotBlendEnd
 - InterpolatedErrorCorrectionSettings, [269](#)
- RotBlendStart
 - InterpolatedErrorCorrectionSettings, [269](#)
- RotTeleportRadians
 - InterpolatedErrorCorrectionSettings, [270](#)
- RoundBitsUpTo32
 - Native, [361](#)
- RoundBitsUpTo64
 - Native, [361](#)
- RoundToAlignment
 - Native, [361](#), [362](#)
- RoundToByte
 - BitStream, [800](#)
- RoundToMaxAlignment
 - Native, [362](#)
- RoundTripTime
 - EngineProfiler, [151](#)
- FusionStatisticsSnapshot, [1083](#)
- NetConnection, [1027](#)
- RoundTripTimeCallback
 - EngineProfiler, [155](#)
- RpcAttribute, [860](#)
 - Channel, [862](#)
 - HostMode, [862](#)
 - InvokeLocal, [863](#)
 - MaxPayloadSize, [862](#)
 - RpcAttribute, [862](#)
 - Sources, [863](#)
 - Targets, [863](#)
 - TickAligned, [863](#)
- RpcChannel
 - Fusion, [53](#)
- RpcHeader, [863](#)
 - Behaviour, [866](#)
 - Create, [864](#), [865](#)
 - Method, [867](#)
 - Object, [867](#)
 - Read, [865](#)
 - ReadSize, [865](#)
 - SIZE, [867](#)
 - ToString, [866](#)
 - Write, [866](#)
- RpcHostMode
 - Fusion, [54](#)
- RpcIn
 - EngineProfiler, [151](#)
- RpcInCallback
 - EngineProfiler, [155](#)
- RpcInfo, [867](#)
 - Channel, [869](#)
 - FromLocal, [868](#)
 - FromMessage, [868](#)
 - IsInvokeLocal, [869](#)
 - Source, [869](#)
 - Tick, [869](#)
 - ToString, [869](#)
- RpcInvokeData, [870](#)
 - Delegate, [871](#)
 - Key, [871](#)
 - Sources, [871](#)
 - Targets, [871](#)
 - ToString, [870](#)
- RpcInvokeDelegate
 - Fusion, [61](#)
- RpcInvokeInfo, [871](#)
 - LocalInvokeResult, [872](#)
 - SendCullResult, [872](#)
 - SendResult, [872](#)
 - ToString, [872](#)
- RpcLocalInvokeResult
 - Fusion, [54](#)
- RpcOut
 - EngineProfiler, [151](#)
- RpcOutCallback
 - EngineProfiler, [155](#)

- RpcSendCullResult
 - Fusion, [54](#)
- RpcSendMessageResult
 - Fusion, [56](#)
- RpcSendResult, [873](#)
 - MessageSize, [873](#)
 - Result, [874](#)
 - ToString, [873](#)
- RpcSources
 - Fusion, [56](#)
- RpcStaticInvokeDelegate
 - Fusion, [62](#)
- RpcTargetAttribute, [874](#)
 - RpcTargetAttribute, [874](#)
- RpcTargets
 - Fusion, [57](#)
- RpcTargetStatus
 - Fusion, [57](#)
- Run
 - TaskManager, [103](#)
- Runner
 - NetworkObject, [538](#)
 - NetworkSpawnOp, [729](#)
 - SimulationBehaviour, [927](#)
- Running
 - NetworkRunner, [641](#)
 - TickAccumulator, [1104](#)
- RuntimeUnityFlagsSetup, [875](#)
 - Check_ENABLE_IL2CPP, [875](#)
 - Check_ENABLE_MONO, [875](#)
 - Check_NET_4_6, [876](#)
 - Check_NET_STANDARD_2_0, [876](#)
 - Check_NETFX_CORE, [876](#)
 - Check_UNITY_2019_4_OR_NEWER, [876](#)
 - Check_UNITY_EDITOR, [876](#)
 - Check_UNITY_GAMECORE, [876](#)
 - Check_UNITY_SWITCH, [877](#)
 - Check_UNITY_WEBGL, [877](#)
 - Check_UNITY_XBOXONE, [877](#)
 - Reset, [877](#)
- SampleWindowSeconds
 - TimeSyncConfiguration, [1124](#)
- Saw
 - NetConfigSimulationOscillator, [1024](#)
- Scale
 - NetworkTRSPData, [767](#)
- Scene
 - NetworkSceneObjectId, [723](#)
 - SceneLoadDoneArgs, [878](#)
 - StartGameArgs, [1074](#)
- SceneCount
 - NetworkSceneInfo, [716](#)
- SceneCountMask
 - Fusion, [51](#)
- SceneLoadDone
 - ISceneLoadDone, [272](#)
- SceneLoadDoneArgs, [877](#)
 - RootGameObjects, [878](#)
 - Scene, [878](#)
 - SceneLoadDoneArgs, [878](#)
 - SceneObjects, [879](#)
 - SceneRef, [879](#)
- SceneLoadId
 - NetworkSceneObjectId, [723](#)
- SceneLoadStart
 - ISceneLoadStart, [273](#)
- SceneManager
 - NetworkRunner, [695](#)
 - StartGameArgs, [1074](#)
- SceneObject
 - Fusion, [52](#)
- SceneObjects
 - SceneLoadDoneArgs, [879](#)
- SceneParams
 - NetworkSceneInfo, [716](#)
- ScenePathAttribute, [879](#)
- SceneRef, [879](#)
 - AsIndex, [886](#)
 - AsPathHash, [886](#)
 - Equals, [881](#)
 - FLAG_ADDRESSABLE, [885](#)
 - FromIndex, [882](#)
 - FromPath, [882](#)
 - FromRaw, [883](#)
 - GetHashCode, [883](#)
 - IsIndex, [886](#)
 - IsPath, [883](#)
 - IsValid, [886](#)
 - NetworkSceneAsyncOp, [709](#)
 - None, [887](#)
 - operator!=, [884](#)
 - operator==, [884](#)
 - Parse, [884](#)
 - RawValue, [885](#)
 - SceneLoadDoneArgs, [879](#)
 - SIZE, [886](#)
 - ToString, [885](#)
- Scenes
 - NetworkSceneInfo, [716](#)
- Scheduling
 - NetworkProjectConfig, [613](#)
- SchedulingAndInterestManagement
 - NetworkProjectConfig, [613](#)
- SchedulingEnabled
 - SimulationConfig, [933](#)
- SchedulingWithoutAOI
 - SimulationConfig, [933](#)
- ScriptHeaderBackColor
 - Fusion, [57](#)
- ScriptHeaderIcon
 - Fusion, [57](#)
- ScriptHeaderStyle
 - Fusion, [58](#)
- ScriptHelpAttribute, [887](#)
 - BackColor, [887](#)
 - Hide, [887](#)

- Style, [888](#)
- Url, [888](#)
- Seconds
 - Fusion, [60](#)
- SecondsToMicroseconds
 - Maths, [330](#)
- SecondsToMilliseconds
 - Maths, [331](#)
- SeekToByteBoundary
 - NetBitBuffer, [988](#)
- Select< T >
 - NetworkBehaviourBufferInterpolator, [429](#), [430](#)
- Self
 - Fusion, [57](#)
- Send
 - INetSocket, [967](#)
 - NetPeer, [1043](#)
- SendCullResult
 - RpclInvokeInfo, [872](#)
- SendDelta
 - Simulation, [920](#)
- SendMessage
 - Communicator, [826](#)
- SendNotifyDataBuffer
 - NetPeerGroup, [1053](#)
- SendPackage
 - Communicator, [827](#)
- SendRate
 - Simulation, [920](#)
- SendReliable
 - NetPeerGroup, [1053](#)
- SendReliableDataToPlayer
 - NetworkRunner, [664](#)
- SendReliableDataToServer
 - NetworkRunner, [665](#)
- SendResult
 - RpclInvokeInfo, [872](#)
- SendRpc
 - NetworkRunner, [665](#)
- SendTime
 - NetSendEnvelope, [1056](#)
- SendUnconnectedData
 - NetPeerGroup, [1053](#)
- SendUnreliableDataBuffer
 - NetPeerGroup, [1054](#)
- SendWindowFull
 - Fusion.Sockets, [69](#)
- Sent
 - SimulationInput, [935](#)
- SentBroadcast
 - Fusion, [56](#)
- SentToServerForForwarding
 - Fusion, [56](#)
- SentToTargetClient
 - Fusion, [56](#)
- Sequence
 - NetSendEnvelope, [1056](#)
 - ReliableId, [1061](#)
- SequenceBounds
 - NetConfigNotify, [1021](#)
- SequenceBytes
 - NetConfigNotify, [1020](#)
- SequenceOutOfBounds
 - Fusion.Sockets, [69](#)
- SerializableDictionary< TKey, TValue >, [888](#)
 - Add, [889](#)
 - Clear, [891](#)
 - ContainsKey, [891](#)
 - Count, [894](#)
 - Create< TKey, TValue >, [891](#)
 - EntryKeyPropertyPath, [893](#)
 - GetEnumerator, [891](#)
 - IsReadOnly, [894](#)
 - ItemsPropertyPath, [893](#)
 - Keys, [894](#)
 - Remove, [892](#)
 - Reset, [892](#)
 - Store, [892](#)
 - this[TKey key], [894](#)
 - TryGetValue, [892](#)
 - Values, [894](#)
 - Wrap, [893](#)
- SerializableType
 - SerializableType< BaseType >, [896](#)
- SerializableType< BaseType >, [895](#)
 - AssemblyQualifiedName, [898](#)
 - AsShort, [897](#)
 - Equals, [897](#)
 - GetHashCode, [897](#)
 - GetShortAssemblyQualifiedName, [897](#)
 - IsValid, [898](#)
 - operator SerializableType, [897](#)
 - operator Type, [898](#)
 - SerializableType, [896](#)
 - Value, [898](#)
- SerializableTypeAttribute, [899](#)
 - BaseType, [899](#)
 - UseFullAssemblyQualifiedName, [899](#)
 - WarnIfNoPreserveAttribute, [899](#)
- Serialize
 - BitStream, [800–804](#), [806–808](#)
 - NetBitBufferSerializer, [1008–1010](#)
 - NetworkProjectConfig, [614](#)
- SerializeArray< T >
 - BitStream, [808](#)
- SerializeArrayLength< T >
 - BitStream, [809](#)
- SerializeBuffer
 - BitStream, [809–811](#)
- SerializeReferenceTypePickerAttribute, [900](#)
 - GroupTypesByNamespace, [901](#)
 - SerializeReferenceTypePickerAttribute, [900](#)
 - ShowFullName, [901](#)
 - Types, [901](#)
- Server
 - Fusion, [48](#), [59](#)

- TickRate.Resolved, [1113](#)
- ServerAlreadyInRoom
 - Fusion.Protocol, [66](#)
- ServerFull
 - Fusion.Sockets, [69](#)
- ServerIndex
 - TickRate.Selection, [1115](#)
- ServerIndexOutOfRange
 - TickRate, [1106](#)
- ServerInRoom
 - Fusion, [59](#)
- ServerLogic
 - Fusion.Protocol, [66](#)
- ServerMode
 - SimulationRuntimeConfig, [954](#)
- ServerRefused
 - Fusion.Sockets, [69](#)
- ServerSend
 - TickRate.Resolved, [1113](#)
- ServerSendDelta
 - TickRate.Resolved, [1114](#)
- ServerSendIndex
 - TickRate.Selection, [1116](#)
- ServerSendIndexOutOfRange
 - TickRate, [1106](#)
- ServerSendRateLargerThanTickRate
 - TickRate, [1106](#)
- ServerTickDelta
 - TickRate.Resolved, [1114](#)
- ServerTickStride
 - TickRate.Resolved, [1114](#)
- Service
 - ICommunicator, [827](#)
 - TaskManager, [104](#)
- SessionInfo, [901](#)
 - IsOpen, [903](#)
 - IsValid, [903](#)
 - IsVisible, [903](#)
 - MaxPlayers, [903](#)
 - Name, [904](#)
 - NetworkRunner, [696](#)
 - operator bool, [902](#)
 - PlayerCount, [904](#)
 - Properties, [904](#)
 - Region, [904](#)
 - ToString, [902](#)
 - UpdateCustomProperties, [903](#)
- SessionLobby
 - Fusion, [58](#)
- SessionName
 - StartGameArgs, [1074](#)
- SessionNameGenerator
 - StartGameArgs, [1074](#)
- SessionProperties
 - StartGameArgs, [1074](#)
- Set
 - NetworkArray< T >, [377](#)
 - NetworkButtons, [463](#)
 - NetworkDictionary< K, V >, [478](#)
 - NetworkLinkedList< T >, [512](#)
 - NetworkString< TSize >, [753](#)
- Set< T >
 - NetworkButtons, [464](#)
 - NetworkInput, [503](#)
- SetAllDown
 - NetworkButtons, [464](#)
- SetAllUp
 - NetworkButtons, [464](#)
- SetAreaOfInterestCellSize
 - NetworkRunner, [665](#)
- SetAreaOfInterestGrid
 - NetworkRunner, [666](#)
- SetAreaOfInterestOverride
 - NetworkTransform, [760](#)
 - NetworkTRSP, [764](#)
- SetBehaviourReplicateTo
 - NetworkRunner, [666](#)
- SetBehaviourReplicateToAll
 - NetworkRunner, [667](#)
- SetBit
 - Mask256, [316](#)
- SetBuffer
 - BitStream, [812](#)
- SetBufferLengthBytes
 - NetBitBuffer, [988](#)
- SetCurrentVersion
 - NetworkObjectFlagsExtensions, [540](#)
- SetDown
 - NetworkButtons, [464](#)
- SetDown< T >
 - NetworkButtons, [465](#)
- SetDummy
 - SimulationMessage, [947](#)
- SetForcedAlive< T >
 - DynamicHeap, [134](#)
- SetHitboxActive
 - HitboxRoot, [210](#)
- SetIgnored
 - NetworkObjectFlagsExtensions, [540](#)
- SetIsSimulated
 - NetworkRunner, [667](#)
- SetLayer
 - Hitbox, [191](#)
- SetMasterClient
 - NetworkRunner, [667](#)
- SetMinBoundingRadius
 - HitboxRoot, [211](#)
- SetNotTickAligned
 - SimulationMessage, [947](#)
- SetParentTransform
 - NetworkTRSP, [764](#)
- SetPlayerAlwaysInterested
 - NetworkObject, [533](#)
 - NetworkRunner, [668](#)
- SetPlayerObject
 - NetworkRunner, [668](#)

- SetSimulateMultiPeerPhysics
 - NetworkRunner, 668
- SetStatic
 - SimulationMessage, 947
- SetTarget
 - SimulationMessage, 947
- SetTrigger
 - NetworkMecanimAnimator, 525
- SetUnreliable
 - SimulationMessage, 948
- SetUp
 - NetworkButtons, 465
- Setup
 - IDataEncryption, 145
 - TaskManager, 106
- SetUp< T >
 - NetworkButtons, 465
- SetupEncryption
 - INetSocket, 968
- Shape
 - NetConfigSimulationOscillator, 1025
- Shared
 - Fusion, 48, 58, 60
- SharedModeStateAuthLocalPlayer
 - Fusion, 51
- SharedModeStateAuthMasterClient
 - Fusion, 51
- ShortVersion
 - Versioning, 1156
- ShouldRegisterRpcInvokeDelegates
 - NetworkBehaviourUtils, 450
- ShowFlagsButtons
 - ExpandableEnumAttribute, 158
- ShowFullName
 - SerializeReferenceTypePickerAttribute, 901
- ShowInlineHelp
 - ExpandableEnumAttribute, 158
- ShowTypeHelp
 - InlineHelpAttribute, 255
- Shutdown
 - Fusion.Sockets, 69
 - INetworkObjectProvider, 240
 - INetworkRunnerUpdater, 249
 - INetworkSceneManager, 252
 - NetworkRunner, 641, 669
- ShutdownReason
 - Fusion, 58
 - StartGameResult, 1077
- Simulation, 904
 - ActivePlayers, 915
 - AfterSimulation, 908
 - AfterUpdate, 908
 - BeforeFirstTick, 908
 - BeforeSimulation, 908
 - BeforeUpdate, 908
 - Config, 915
 - DeltaTime, 915
 - GetAreaOfInterestGizmoData, 908
 - GetInputAuthority, 909
 - GetInputForPlayer, 909
 - GetObjectsAndPlayersInAreaOfInterestCell, 909
 - GetObjectsInAreaOfInterestForPlayer, 909
 - GetStateAuthority, 910
 - HasAnyActiveConnections, 910
 - InputCount, 915
 - IsClient, 916
 - IsFirstTick, 916
 - IsForward, 916
 - IsInputAuthority, 910
 - IsInterestedIn, 911
 - IsLastTick, 916
 - IsLocalPlayerFirstExecution, 916
 - IsLocalSimulationInputAuthority, 911
 - IsLocalSimulationStateAuthority, 911, 912
 - IsLocalSimulationStateOrInputSource, 912
 - IsMasterClient, 917
 - IsPlayer, 917
 - IsResimulation, 917
 - IsRunning, 917
 - IsServer, 917
 - IsShutdown, 917
 - IsSinglePlayer, 918
 - IsStateAuthority, 913
 - LatestServerTick, 918
 - LocalAddress, 918
 - LocalAlpha, 918
 - LocalPlayer, 918
 - Mode, 918
 - NetConfig, 1018
 - NetConfigPointer, 919
 - NetworkConnected, 913
 - NetworkDisconnected, 914
 - NetworkProjectConfig, 619
 - NetworkReceiveDone, 914
 - NoSimulation, 914
 - ObjectCount, 919
 - Objects, 919
 - ProjectConfig, 919
 - RemoteAlpha, 919
 - RemoteTick, 919
 - RemoteTickPrevious, 920
 - SendDelta, 920
 - SendRate, 920
 - Stage, 920
 - Tick, 920
 - TickDeltaDouble, 920
 - TickDeltaFloat, 921
 - TickPrevious, 921
 - TickRate, 921
 - TickStride, 921
 - Time, 921
 - Topology, 921
 - TryGetHostPlayer, 914
 - Update, 915
- Simulation.AreaOfInterest, 922
- CELL_SIZE, 924

- GetCellSize, [922](#)
- SphereToCells, [923](#)
- ToCell, [923](#)
- ToCellCenter, [924](#)
- x, [924](#)
- SimulationBehaviour, [925](#)
 - CanReceiveRenderCallback, [927](#)
 - CanReceiveSimulationCallback, [927](#)
 - FixedUpdateNetwork, [926](#)
 - Object, [927](#)
 - Render, [926](#)
 - Runner, [927](#)
- SimulationBehaviourAttribute, [927](#)
 - Modes, [928](#)
 - Stages, [928](#)
 - Topologies, [928](#)
- SimulationBehaviourListScope, [929](#)
 - Dispose, [929](#)
- SimulationConfig, [929](#)
 - AreaOfInterestEnabled, [933](#)
 - DataConsistency, [930](#)
 - DeltaTime, [931](#)
 - Eventual, [931](#)
 - Full, [931](#)
 - HostMigration, [931](#)
 - InputDataWordCount, [931](#)
 - InputTotalWordCount, [933](#)
 - InputTransferMode, [932](#)
 - InputTransferModes, [931](#)
 - LatestState, [931](#)
 - ObjectDataConsistency, [932](#)
 - PlayerCount, [932](#)
 - Redundancy, [931](#)
 - RedundancyUncompressed, [931](#)
 - ReplicationFeatures, [932](#)
 - SchedulingEnabled, [933](#)
 - SchedulingWithoutAOI, [933](#)
 - SimulationTimeMode, [931](#)
 - SimulationUpdateTimeMode, [932](#)
 - TickRateSelection, [932](#)
 - Topology, [933](#)
 - UnscaledDeltaTime, [931](#)
- SimulationEnter
 - ISimulationEnter, [274](#)
- SimulationExit
 - ISimulationExit, [274](#)
- SimulationInput, [934](#)
 - Clear, [934](#)
 - CopyFrom, [935](#)
 - Data, [935](#)
 - Header, [935](#)
 - Player, [935](#)
 - Sent, [935](#)
- SimulationInput.Buffer, [936](#)
 - Add, [937](#)
 - Buffer, [936](#)
 - Clear, [937](#)
 - Contains, [937](#)
 - CopySortedTo, [938](#)
 - Count, [939](#)
 - Full, [939](#)
 - Get, [938](#)
 - GetInsertTime, [938](#)
 - GetLastUsedInputHeader, [939](#)
 - Remove, [939](#)
- SimulationInputHeader, [940](#)
 - InterpAlpha, [940](#)
 - InterpFrom, [940](#)
 - InterpTo, [941](#)
 - SIZE, [941](#)
 - Tick, [941](#)
 - WORD_COUNT, [941](#)
- SimulationMessage, [941](#)
 - Allocate, [944](#)
 - CanAllocateUserPayload, [944](#)
 - Capacity, [949](#)
 - Clone, [944](#)
 - FLAG_DUMMY, [949](#)
 - FLAG_INTERNAL, [949](#)
 - FLAG_NOT_TICK_ALIGNED, [950](#)
 - FLAG_REMOTE, [950](#)
 - FLAG_STATIC, [950](#)
 - FLAG_TARGET_PLAYER, [950](#)
 - FLAG_TARGET_SERVER, [950](#)
 - FLAG_UNRELIABLE, [950](#)
 - FLAG_USER_FLAGS_START, [951](#)
 - FLAG_USER_MESSAGE, [951](#)
 - Flags, [951](#)
 - FLAGS_RESERVED, [951](#)
 - FLAGS_RESERVED_BITS, [951](#)
 - GetData, [945](#)
 - GetFlag, [945](#)
 - IsTargeted, [945](#)
 - IsUnreliable, [953](#)
 - MAX_PAYLOAD_SIZE, [951](#)
 - Offset, [952](#)
 - ReadInt, [946](#)
 - ReadNetworkedObjectRef, [946](#)
 - ReadVector3, [946](#)
 - ReferenceCountAdd, [947](#)
 - ReferenceCountSub, [947](#)
 - References, [952](#)
 - SetDummy, [947](#)
 - SetNotTickAligned, [947](#)
 - SetStatic, [947](#)
 - SetTarget, [947](#)
 - SetUnreliable, [948](#)
 - SIZE, [952](#)
 - Source, [952](#)
 - Target, [952](#)
 - Tick, [952](#)
 - ToString, [948](#)
 - WriteInt, [948](#)
 - WriteNetworkedObjectRef, [948](#)
 - WriteVector3, [949](#)
- SimulationMessagePtr, [953](#)

- Message, [953](#)
- SimulationModes
 - Fusion, [59](#)
- SimulationOffset
 - EngineProfiler, [152](#)
- SimulationOffsetCallback
 - EngineProfiler, [155](#)
- SimulationOffsetDeviation
 - EngineProfiler, [152](#)
- SimulationOffsetDeviationCallback
 - EngineProfiler, [155](#)
- SimulationRuntimeConfig, [953](#)
 - HostPlayer, [954](#)
 - MasterClient, [954](#)
 - PlayerMaxCount, [954](#)
 - ServerMode, [954](#)
 - TickRate, [955](#)
 - Topology, [955](#)
- SimulationSpeed
 - EngineProfiler, [152](#)
 - FusionStatisticsSnapshot, [1084](#)
- SimulationSpeedCallback
 - EngineProfiler, [156](#)
- SimulationStages
 - Fusion, [59](#)
- SimulationState
 - NetworkBehaviour.ChangeDetector, [408](#)
- SimulationTime
 - NetworkRunner, [696](#)
- SimulationTimeMode
 - SimulationConfig, [931](#)
- SimulationTimeOffset
 - FusionStatisticsSnapshot, [1084](#)
- SimulationUnityScene
 - NetworkRunner, [696](#)
- SimulationUpdateTimeMode
 - SimulationConfig, [932](#)
- Sine
 - NetConfigSimulationOscillator, [1024](#)
- Single
 - Fusion, [48](#)
 - Maths.FastAbs, [334](#)
 - NetworkProjectConfig, [613](#)
- SinglePlayerContinue
 - NetworkRunner, [669](#)
- SinglePlayerPause
 - NetworkRunner, [669](#)
- SIZE
 - _128, [73](#)
 - _16, [74](#)
 - _2, [75](#)
 - _256, [76](#)
 - _32, [77](#)
 - _4, [78](#)
 - _512, [79](#)
 - _64, [80](#)
 - _8, [80](#)
 - Allocator.Config, [86](#)
- Angle, [99](#)
- NetworkBehaviourId, [437](#)
- NetworkBool, [459](#)
- NetworkId, [500](#)
- NetworkObjectGuid, [549](#)
- NetworkObjectHeader, [556](#)
- NetworkObjectNestingKey, [564](#)
- NetworkObjectTypeId, [578](#)
- NetworkPhysicsInfo, [582](#)
- NetworkPrefabId, [589](#)
- NetworkPrefabRef, [600](#)
- NetworkRRNG, [628](#)
- NetworkSceneInfo, [715](#)
- NetworkTRSPData, [767](#)
- PlayerRef, [777](#)
- Ptr, [832](#)
- ReliableHeader, [1059](#)
- ReliableId, [1061](#)
- ReliableKey, [1064](#)
- RpcHeader, [867](#)
- SceneRef, [886](#)
- SimulationInputHeader, [941](#)
- SimulationMessage, [952](#)
- Tick, [1098](#)
- TickRate.Resolved, [1113](#)
- Size
 - BitStream, [824](#)
- SizeOf
 - Native, [363](#)
- SizeOfBits< T >
 - Maths, [331](#)
- SliceLength
 - ReliableId, [1061](#)
- SnapshotFrom
 - NetworkBehaviour.ChangeDetector, [408](#)
- SnapshotHistoryDraw, [305](#)
 - GetEnumerator, [305](#)
 - LagCompensationDraw, [292](#)
- SnapshotTo
 - NetworkBehaviour.ChangeDetector, [408](#)
- SocketRecvBuffer
 - NetConfig, [1018](#)
- SocketRecvBufferSize
 - NetworkConfiguration, [471](#)
- SocketSendBuffer
 - NetConfig, [1018](#)
- SocketSendBufferSize
 - NetworkConfiguration, [472](#)
- SortDistance
 - LagCompensatedExt, [290](#)
- SortKey
 - NetworkObject, [535](#)
- SortReference
 - LagCompensatedExt, [291](#)
- Source
 - NetworkBehaviour.ChangeDetector, [408](#)
 - ReliableId, [1061](#)
 - RenderAttribute, [858](#)

- RpcInfo, 869
- SimulationMessage, 952
- SourceCombined
 - ReliableId, 1062
- SourcelsHostPlayer
 - Fusion, 54
- SourcelsServer
 - Fusion, 54
- Sources
 - NetworkRpcWeavedInvokerAttribute, 631
 - RpcAttribute, 863
 - RpcInvokeData, 871
- SourceSend
 - ReliableId, 1061
- Spawn
 - NetworkRunner, 669–672
- Spawn< T >
 - NetworkRunner, 672
- SpawnAsync
 - NetworkRunner, 673–675
- SpawnAsync< T >
 - NetworkRunner, 676
- Spawned
 - Fusion, 52
 - ISpawned, 275
 - NetworkBehaviour, 402
- SpawnedByClient
 - Fusion, 50
- Sphere
 - Fusion, 49
- SphereOverlapQuery, 305
 - Center, 307
 - Check, 307
 - Radius, 307
 - SphereOverlapQuery, 306
- SphereOverlapQueryParams, 307
 - Center, 308
 - QueryParams, 309
 - Radius, 309
 - SphereOverlapQueryParams, 308
 - StaticHitsCapacity, 309
- SphereRadius
 - Hitbox, 192
- SphereToCells
 - Simulation.AreaOfInterest, 923
- Square
 - NetConfigSimulationOscillator, 1024
- SquareMagnitude
 - Fusion, 60
- StackTrace
 - StartGameResult, 1077
- Stage
 - NetworkRunner, 696
 - Simulation, 920
- Stages
 - SimulationBehaviourAttribute, 928
- Start
 - TickAccumulator, 1103
 - Timer, 1121
- StartGame
 - NetworkRunner, 677
- StartGameArgs, 1068
 - Address, 1070
 - AuthValues, 1070
 - Config, 1071
 - ConnectionToken, 1071
 - CustomCallbackInterfaces, 1071
 - CustomLobbyName, 1071
 - CustomPhotonAppSettings, 1071
 - CustomPublicAddress, 1071
 - CustomSTUNServer, 1072
 - DisableNATPunchthrough, 1072
 - EnableClientSessionCreation, 1072
 - GameMode, 1072
 - HostMigrationResume, 1072
 - HostMigrationToken, 1072
 - IsOpen, 1073
 - IsVisible, 1073
 - MatchmakingMode, 1073
 - ObjectInitializer, 1073
 - ObjectProvider, 1073
 - OnGameStarted, 1073
 - PlayerCount, 1074
 - Scene, 1074
 - SceneManager, 1074
 - SessionName, 1074
 - SessionNameGenerator, 1074
 - SessionProperties, 1074
 - StartGameCancellationToken, 1075
 - ToString, 1070
 - Updater, 1075
 - UseCachedRegions, 1075
 - UseDefaultPhotonCloudPorts, 1075
- StartGameCancellationToken
 - StartGameArgs, 1075
- StartGameResult, 1075
 - ErrorMessage, 1076
 - Ok, 1076
 - ShutdownReason, 1077
 - StackTrace, 1077
 - ToString, 1076
- Starting
 - NetworkRunner, 641
- StartNew
 - TickAccumulator, 1103
 - Timer, 1121
- StartsWith
 - NetworkString< TSize >, 754
- StartsWith< TOtherSize >
 - NetworkString< TSize >, 754
- STATE
 - AuthorityMasks, 110
- State
 - NetworkRunner, 696
 - NetworkTRSP, 765
- StateAuthority

- Fusion, [57](#)
- NetworkObject, [538](#)
- NetworkObjectHeader, [556](#)
- NetworkObjectMeta, [561](#)
- StateAuthorityChanged
 - IStateAuthorityChanged, [276](#)
- StateBuffer
 - NetworkBehaviour, [404](#)
- StateBufferIsValid
 - NetworkBehaviour, [405](#)
- StateReceiveDelta
 - FusionStatisticsSnapshot, [1084](#)
- StateRecvDelta
 - EngineProfiler, [153](#)
- StateRecvDeltaCallback
 - EngineProfiler, [156](#)
- StateRecvDeltaDeviation
 - EngineProfiler, [153](#)
- StateRecvDeltaDeviationCallback
 - EngineProfiler, [156](#)
- States
 - NetworkRunner, [641](#)
- StaticConstructorAttribute, [87](#)
- StaticFieldAttribute, [88](#)
 - Reset, [89](#)
 - StaticFieldAttribute, [88](#)
- StaticFieldResetMethodAttribute, [89](#)
 - StaticFieldResetMethodAttribute, [89, 90](#)
- StaticFieldResetMode
 - Fusion.Analyzer, [62](#)
- StaticHitsCapacity
 - BoxOverlapQueryParams, [285](#)
 - RaycastQueryParams, [304](#)
 - SphereOverlapQueryParams, [309](#)
- Status
 - NetworkObjectSpawnException, [571](#)
 - NetworkSpawnOp, [730](#)
- Stop
 - TickAccumulator, [1103](#)
 - Timer, [1122](#)
- Store
 - SerializableDictionary< TKey, TValue >, [892](#)
- Struct
 - Fusion, [50](#)
- StructArray
 - Fusion, [50](#)
- StunServers.StunServer, [1068](#)
- Style
 - ScriptHelpAttribute, [888](#)
- Substring
 - NetworkString< TSize >, [755](#)
- SubtickAccuracy
 - Fusion, [49](#)
- Success
 - Fusion, [49, 50](#)
- SupportsMultiThreading
 - INetSocket, [968](#)
- SurrogateType
 - FixedBufferPropertyAttribute, [173](#)
- Symmetric
 - Fusion.Sockets.Stun, [70](#)
- SyncParent
 - NetworkTransform, [761](#)
- SyncScale
 - NetworkTransform, [761](#)
- T
 - NetworkBehaviour.BehaviourReader< T >, [407](#)
 - NetworkBehaviour.PropertyReader< T >, [415](#)
- TargetPlayerIsNotLocal
 - Fusion, [54](#)
- Tail
 - ReliableList, [1067](#)
- Target
 - ReliableId, [1061](#)
 - SimulationMessage, [952](#)
- TargetAllocator
 - MemoryStatisticsSnapshot, [1088](#)
- TargetMethod
 - FieldEditorButtonAttribute, [160](#)
- TargetPlayerIsLocalButRpcsNotInvokableLocally
 - Fusion, [56](#)
- TargetPlayerIsNotLocal
 - Fusion, [54](#)
- TargetPlayerUnreachable
 - Fusion, [56](#)
- Targets
 - NetworkRpcWeavedInvokerAttribute, [631](#)
 - RpcAttribute, [863](#)
 - RpcInvokeData, [871](#)
- TargetTick
 - TickTimer, [1120](#)
- TaskManager, [102](#)
 - ContinueWhenAll, [103](#)
 - Delay, [103](#)
 - Run, [103](#)
 - Service, [104](#)
 - Setup, [106](#)
- Teleport
 - INetworkTRSPTeleport, [254](#)
 - NetworkTransform, [761](#)
 - NetworkTRSP, [764](#)
- TeleportKey
 - NetworkTRSPData, [767](#)
- this[int i]
 - Mask256, [316](#)
- this[int index]
 - FixedArray< T >, [169](#)
 - INetworkArray, [234](#)
 - NetworkArray< T >, [378](#)
 - NetworkArrayReadOnly< T >, [383](#)
 - NetworkBehaviourBuffer, [422](#)
 - NetworkLinkedList< T >, [513](#)
 - NetworkLinkedListReadOnly< T >, [519](#)
 - NetworkString< TSize >, [757](#)
 - TickRate, [1111](#)
- this[K key]

- NetworkDictionary< K, V >, 479
- this[TKey key]
 - SerializableDictionary< TKey, TValue >, 894
- Threshold
 - NetConfigSimulationOscillator, 1026
- ThrowIfBehaviourNotInitialized
 - NetworkBehaviourUtils, 451
- Tick, 1092
 - ALIGNMENT, 1098
 - CompareTo, 1093
 - Equals, 1093, 1094
 - GetHashCode, 1094
 - NetworkBehaviourBuffer, 422
 - NetworkRunner, 696
 - Next, 1094
 - operator bool, 1095
 - operator int, 1095
 - operator Tick, 1095
 - operator!=, 1097
 - operator<, 1097
 - operator<=, 1097
 - operator>, 1097
 - operator>=, 1098
 - operator==, 1097
 - Query, 297
 - QueryParams, 299
 - Raw, 1098
 - RpcInfo, 869
 - Simulation, 920
 - SimulationInputHeader, 941
 - SimulationMessage, 952
 - SIZE, 1098
 - ToString, 1098
- Tick.EqualityComparer, 1099
 - Equals, 1099
 - GetHashCode, 1099
- Tick.RelationalComparer, 1100
 - Compare, 1100
- TickAccumulator, 1101
 - AddTicks, 1102
 - AddTime, 1102
 - Alpha, 1102
 - ConsumeTick, 1103
 - Pending, 1104
 - Remainder, 1104
 - Running, 1104
 - Start, 1103
 - StartNew, 1103
 - Stop, 1103
 - TimeScale, 1104
- TickAligned
 - RpcAttribute, 863
- TickDeltaDouble
 - Simulation, 920
- TickDeltaFloat
 - Simulation, 921
- TickPrevious
 - Simulation, 921
- TickRate, 1105
 - Available, 1111
 - ClampSelection, 1107
 - Client, 1111
 - ClientSendIndexOutOfRange, 1106
 - Count, 1111
 - Error, 1106
 - Get, 1107
 - GetDivisor, 1107
 - GetTickRate, 1108
 - Init, 1108
 - InvalidTickRate, 1106
 - IsValid, 1108, 1109
 - NetworkRunner, 697
 - NotFound, 1106
 - Ok, 1106
 - Resolve, 1109
 - ServerIndexOutOfRange, 1106
 - ServerSendIndexOutOfRange, 1106
 - ServerSendRateLargerThanTickRate, 1106
 - Simulation, 921
 - SimulationRuntimeConfig, 955
 - this[int index], 1111
 - ToArray, 1110
 - Validate, 1110
 - ValidateResult, 1106
 - ValidateSelection, 1110
- TickRate.Resolved, 1112
 - Client, 1113
 - ClientSend, 1113
 - ClientSendDelta, 1114
 - ClientTickDelta, 1114
 - ClientTickStride, 1114
 - Server, 1113
 - ServerSend, 1113
 - ServerSendDelta, 1114
 - ServerTickDelta, 1114
 - ServerTickStride, 1114
 - SIZE, 1113
 - WORDS, 1113
- TickRate.Selection, 1115
 - Client, 1115
 - ClientSendIndex, 1115
 - ServerIndex, 1115
 - ServerSendIndex, 1116
- TickRateSelection
 - SimulationConfig, 932
- Ticks
 - Fusion, 60
- TicksExecuted
 - NetworkRunner, 697
- TicksPerSecond
 - Fusion, 60
- TickStride
 - Simulation, 921
- TickTimer, 1116
 - CreateFromSeconds, 1117
 - CreateFromTicks, 1117

- Expired, [1118](#)
- ExpiredOrNotRunning, [1118](#)
- IsRunning, [1119](#)
- None, [1120](#)
- RemainingTicks, [1118](#)
- RemainingTime, [1119](#)
- TargetTick, [1120](#)
- ToString, [1119](#)
- TickTo
 - Query, [297](#)
 - QueryParams, [299](#)
- Time
 - NetPeerGroup, [1055](#)
 - Simulation, [921](#)
- Timeframe
 - RenderAttribute, [858](#)
- Timeout
 - Fusion.Sockets, [69](#)
- Timer, [1120](#)
 - ElapsedInMilliseconds, [1122](#)
 - ElapsedInSeconds, [1122](#)
 - ElapsedInTicks, [1122](#)
 - IsRunning, [1122](#)
 - Reset, [1121](#)
 - Restart, [1121](#)
 - Start, [1121](#)
 - StartNew, [1121](#)
 - Stop, [1122](#)
- TimeResets
 - FusionStatisticsSnapshot, [1084](#)
- TimeScale
 - NetworkPhysicsInfo, [582](#)
 - TickAccumulator, [1104](#)
- TimeSyncConfiguration, [1123](#)
 - MaxLateInputs, [1123](#)
 - MaxLateSnapshots, [1123](#)
 - RedundantInputs, [1123](#)
 - RedundantSnapshots, [1124](#)
 - SampleWindowSeconds, [1124](#)
- TimeSynchronizationOverride
 - NetworkProjectConfig, [619](#)
- To
 - Fusion, [53](#)
 - NetworkBehaviourBufferInterpolator, [433](#)
- ToArray
 - BitStream, [812](#)
 - FixedArray< T >, [169](#)
 - NetworkArray< T >, [377](#)
 - TickRate, [1110](#)
- ToCell
 - Simulation.AreaOfInterest, [923](#)
- ToCellCenter
 - Simulation.AreaOfInterest, [924](#)
- ToggleLeftAttribute, [1124](#)
- ToListString
 - FixedArray< T >, [169](#)
 - NetworkArray< T >, [378](#)
- ToLower
 - NetworkString< TSize >, [756](#)
- ToNamePrefixString
 - NetworkId, [498](#)
- Topologies
 - Fusion, [60](#)
 - SimulationBehaviourAttribute, [928](#)
- Topology
 - NetworkRunner, [697](#)
 - Simulation, [921](#)
 - SimulationConfig, [933](#)
 - SimulationRuntimeConfig, [955](#)
- ToReadOnly
 - NetworkArray< T >, [378](#)
 - NetworkDictionary< K, V >, [478](#)
- ToString
 - Allocator.Config, [85](#)
 - Angle, [99](#)
 - FixedArray< T >, [169](#)
 - HeapConfiguration, [188](#)
 - Mask256, [316](#)
 - NetAddress, [972](#)
 - NetConnection, [1026](#)
 - NetworkArray< T >, [378](#)
 - NetworkBehaviourId, [436](#)
 - NetworkBool, [459](#)
 - NetworkId, [499](#)
 - NetworkLoadSceneParameters, [522](#)
 - NetworkObjectGuid, [547](#), [548](#)
 - NetworkObjectHeader, [554](#)
 - NetworkObjectNestingKey, [564](#)
 - NetworkObjectReleaseContext, [568](#)
 - NetworkObjectTypeId, [577](#)
 - NetworkPrefabId, [588](#), [589](#)
 - NetworkPrefabRef, [598](#), [599](#)
 - NetworkProjectConfig, [614](#)
 - NetworkRNG, [628](#)
 - NetworkRunnerUpdaterDefaultInvokeSettings, [704](#)
 - NetworkSceneInfo, [715](#)
 - NetworkSceneLoadId, [719](#)
 - NetworkSceneObjectId, [722](#)
 - NetworkString< TSize >, [756](#)
 - PlayerRef, [776](#)
 - Ptr, [832](#)
 - RpcHeader, [866](#)
 - RpcInfo, [869](#)
 - RpcInvokeData, [870](#)
 - RpcInvokeInfo, [872](#)
 - RpcSendResult, [873](#)
 - SceneRef, [885](#)
 - SessionInfo, [902](#)
 - SimulationMessage, [948](#)
 - StartGameArgs, [1070](#)
 - StartGameResult, [1076](#)
 - Tick, [1098](#)
 - TickTimer, [1119](#)
- TotalElapsedTime
 - LagCompensationStatisticsSnapshot, [1087](#)
- TotalFreeBlocks

- MemoryStatisticsSnapshot, [1089](#)
- TotalHitboxes
 - HitboxManager, [207](#)
- TotalLength
 - ReliableId, [1062](#)
- ToUnityGuidString
 - NetworkObjectGuid, [548](#)
 - NetworkPrefabRef, [599](#)
- ToUpper
 - NetworkString< TSize >, [756](#)
- Triangle
 - NetConfigSimulationOscillator, [1024](#)
- TriggerInteraction
 - Query, [297](#)
 - QueryParams, [299](#)
- TryAddSource
 - NetworkPrefabTable, [607](#)
- TryFindBehaviour
 - NetworkRunner, [677](#)
- TryFindBehaviour< T >
 - NetworkRunner, [678](#)
- TryFindByIndex
 - NetConnectionMap, [1035](#)
- TryFindObject
 - NetworkRunner, [678](#)
- TryGet
 - NetworkDictionary< K, V >, [478](#)
 - NetworkDictionaryReadOnly< K, V >, [483](#)
- TryGet< T >
 - NetworkInput, [503](#)
- TryGetBehaviour< T >
 - Behaviour, [112](#)
- TryGetBehaviourStatistics
 - NetworkRunner, [679](#)
- TryGetConnectionByIndex
 - NetPeerGroup, [1054](#)
- TryGetFusionStatistics
 - NetworkRunner, [679](#)
- TryGetGlobal
 - NetworkProjectConfigAsset, [621](#)
- TryGetGlobalInternal
 - FusionGlobalScriptableObject< T >, [180](#)
- TryGetHostPlayer
 - Simulation, [914](#)
- TryGetInputForPlayer< T >
 - NetworkRunner, [679](#)
- TryGetNetworkedBehaviourFromNetworkedObjectRef< T >
 - NetworkRunner, [680](#)
- TryGetNetworkedBehaviourId
 - NetworkRunner, [680](#)
- TryGetObjectRefFromNetworkedBehaviour
 - NetworkRunner, [681](#)
- TryGetPhysicsInfo
 - NetworkRunner, [681](#)
- TryGetPhysicsScene2D
 - INetworkSceneManager, [252](#)
- TryGetPhysicsScene3D
 - INetworkSceneManager, [252](#)
- TryGetPlayerObject
 - NetworkRunner, [681](#)
- TryGetRpcInvokeDelegateArray
 - NetworkBehaviourUtils, [451](#)
- TryGetRpcStaticInvokeDelegate
 - NetworkBehaviourUtils, [451](#)
- TryGetSceneInfo
 - NetworkRunner, [682](#)
- TryGetSnapshotsBuffers
 - NetworkBehaviour, [403](#)
- TryGetValue
 - SerializableDictionary< TKey, TValue >, [892](#)
- TryParse
 - NetworkObjectGuid, [548](#)
 - NetworkPrefabRef, [599](#)
- TrySet< T >
 - NetworkInput, [503](#)
- TrySetPhysicsInfo
 - NetworkRunner, [682](#)
- TrySpawn
 - NetworkRunner, [683](#), [685](#), [686](#)
- TrySpawn< T >
 - NetworkRunner, [687](#)
- Type
 - ColliderDrawInfo, [289](#)
 - FixedBufferPropertyAttribute, [173](#)
 - Hitbox, [192](#)
 - LagCompensatedHit, [279](#)
 - NetworkInput, [504](#)
 - NetworkObjectHeader, [556](#)
 - NetworkObjectHeaderPtr, [558](#)
 - NetworkObjectMeta, [561](#)
 - NetworkProjectConfigAsset.SerializableSimulationBehaviourMeta, [623](#)
- TypeId
 - NetworkObjectReleaseContext, [569](#)
 - NetworkObjectSpawnException, [571](#)
 - NetworkProjectConfig, [619](#)
- Types
 - SerializeReferenceTypePickerAttribute, [901](#)
- UdpBlocked
 - Fusion.Sockets.Stun, [70](#)
- UInt32
 - Maths.FastAbs, [334](#)
- Unit
 - UnitAttribute, [1125](#)
- UnitAttribute, [1124](#)
 - Unit, [1125](#)
 - UnitAttribute, [1125](#)
- Units
 - Fusion, [60](#)
- Unity
 - Fusion, [58](#)
- UnityAddressablesRuntimeKeyAttribute, [1125](#)
- UnityArraySurrogate< T, ReaderWriter >, [257](#)
 - DataProperty, [259](#)
 - Init, [258](#)

- Read, [258](#)
- Write, [258](#)
- UnityAssetGuidAttribute, [1126](#)
- UnityContextMenuMenuItemAttribute, [1126](#)
 - order, [1127](#)
 - UnityContextMenuMenuItemAttribute, [1126](#)
- UnityDelayedAttribute, [1127](#)
 - order, [1127](#)
- UnityDictionarySurrogate< TKeyType, TKeyReader-Writer, TValueType, TValueReaderWriter >, [259](#)
 - DataProperty, [261](#)
 - Init, [260](#)
 - Read, [260](#)
 - Write, [261](#)
- UnityEngine, [71](#)
- UnityEngine.SceneManagement, [71](#)
- UnityEngine.Scripting, [71](#)
- UnityEngine.Serialization, [71](#)
- UnityFormerlySerializedAsAttribute, [1128](#)
 - UnityFormerlySerializedAsAttribute, [1128](#)
- UnityHeaderAttribute, [1128](#)
 - order, [1129](#)
 - UnityHeaderAttribute, [1129](#)
- UnityLinkedListSurrogate< T, ReaderWriter >, [261](#)
 - DataProperty, [263](#)
 - Init, [262](#)
 - Read, [262](#)
 - Write, [263](#)
- UnityMinAttribute, [1129](#)
 - order, [1130](#)
 - UnityMinAttribute, [1130](#)
- UnityMultilineAttribute, [1130](#)
 - order, [1130](#)
- UnityNonReorderableAttribute, [1131](#)
 - order, [1131](#)
- UnityNonSerializedAttribute, [1131](#)
- UnityPlayerLoopSystemAddMode
 - Fusion, [61](#)
- UnityRangeAttribute, [1132](#)
 - order, [1132](#)
 - UnityRangeAttribute, [1132](#)
- UnityResourcePathAttribute, [1133](#)
 - ResourceType, [1133](#)
 - UnityResourcePathAttribute, [1133](#)
- UnitySerializeField, [1134](#)
- UnitySerializeReference, [1134](#)
- UnitySpaceAttribute, [1134](#)
 - order, [1135](#)
 - UnitySpaceAttribute, [1135](#)
- UnitySurrogateBase, [263](#)
 - Init, [264](#)
 - Read, [264](#)
 - Write, [265](#)
- UnityTooltipAttribute, [1135](#)
 - order, [1136](#)
 - UnityTooltipAttribute, [1135](#)
- UnityValueSurrogate< T, TReaderWriter >, [265](#)
 - DataProperty, [267](#)
 - Init, [266](#)
 - Read, [266](#)
 - Write, [266](#)
- Unknown
 - Fusion.Sockets, [69](#)
- Unload
 - NetworkPrefabTable, [608](#)
- UnloadAll
 - NetworkPrefabTable, [608](#)
- Unloader
 - FusionGlobalScriptableObjectLoadResult, [184](#)
- UnloadGlobal
 - NetworkProjectConfig, [615](#)
 - NetworkProjectConfigAsset, [621](#)
- UnloadGlobalInternal
 - FusionGlobalScriptableObject< T >, [181](#)
- UnloadPrefabOnReleasingLastInstance
 - NetworkPrefabTableOptions, [610](#)
- UnloadScene
 - INetworkSceneManager, [252](#)
 - NetworkRunner, [687](#)
- UnloadUnreferenced
 - NetworkPrefabTable, [608](#)
- UnloadUnusedPrefabsOnShutdown
 - NetworkPrefabTableOptions, [610](#)
- Unreachable
 - Fusion, [57](#)
- UnregisterFromPlayerLoop
 - NetworkRunnerUpdaterDefault, [700](#)
- Unreliable
 - Fusion, [54](#)
- unsafe
 - BinUtils, [115](#)
- UnscaledDeltaTime
 - SimulationConfig, [931](#)
- Update
 - NetPeer, [1044](#)
 - NetPeerGroup, [1055](#)
 - Simulation, [915](#)
- UpdateBufferTime
 - LagCompensationStatisticsSnapshot, [1087](#)
- UpdateBVHTime
 - LagCompensationStatisticsSnapshot, [1087](#)
- UpdateCustomProperties
 - SessionInfo, [903](#)
- UpdateDelay
 - HostMigrationConfig, [213](#)
- UpdateInternal
 - NetworkRunner, [689](#)
- Updater
 - StartGameArgs, [1075](#)
- UpdateSettings
 - NetworkRunnerUpdaterDefault, [701](#)
- Url
 - ScriptHelpAttribute, [888](#)
- UseCachedRegions
 - StartGameArgs, [1075](#)

- Used
 - NetConnectionMap, 1032
- UseDefaultPhotonCloudPorts
 - StartGameArgs, 1075
- UseFullAssemblyQualifiedName
 - SerializableTypeAttribute, 899
- UserArgs
 - Query, 297
 - QueryParams, 299
- UserData
 - NetSendEnvelope, 1056
- UserId
 - NetworkRunner, 697
- UseSerializableDictionary
 - NetworkProjectConfig, 619
- UseSlider
 - RangeExAttribute, 840
- UTF32Tools, 1136
 - Convert, 1136, 1137
 - GetLength, 1137
- UTF32Tools.CharEnumerator, 1138
 - Current, 1139
 - Dispose, 1138
 - MoveNext, 1138
 - Reset, 1139
- UTF32Tools.ConversionResult, 1139
 - CharacterCount, 1140
 - CodePointCount, 1140
 - ConversionResult, 1140
- V1
 - Fusion, 50
- Valid
 - NetworkBehaviourBuffer, 423
 - NetworkBehaviourBufferInterpolator, 433
- Validate
 - TickRate, 1110
- ValidateResult
 - TickRate, 1106
- ValidateSelection
 - TickRate, 1110
- Value
 - AtomicInt, 109
 - NetworkObjectNestingKey, 564
 - NetworkSceneLoadId, 720
 - NetworkString< TSize >, 757
 - SerializableType< BaseType >, 898
- valueEncoded
 - FloatCompressed, 177
- Values
 - SerializableDictionary< TKey, TValue >, 894
- ValueWordCount
 - NetworkedWeavedDictionaryAttribute, 488
- Vector2
 - NetworkBehaviourBufferInterpolator, 430, 431
- Vector2Compressed, 1140
 - Equals, 1141, 1142
 - GetHashCode, 1142
 - operator Vector2, 1142
 - operator Vector2Compressed, 1143
 - operator !=, 1143
 - operator ==, 1143
 - X, 1144
 - xEncoded, 1144
 - Y, 1144
 - yEncoded, 1144
- Vector3
 - NetworkBehaviourBufferInterpolator, 431, 432
- Vector3Compressed, 1145
 - Equals, 1146
 - GetHashCode, 1147
 - operator Vector2, 1147
 - operator Vector3, 1147
 - operator Vector3Compressed, 1148
 - operator !=, 1148
 - operator ==, 1149
 - X, 1150
 - xEncoded, 1149
 - Y, 1150
 - yEncoded, 1149
 - Z, 1150
 - zEncoded, 1149
- Vector4
 - NetworkBehaviourBufferInterpolator, 432
- Vector4Compressed, 1150
 - Equals, 1151, 1152
 - GetHashCode, 1152
 - operator Vector4, 1152
 - operator Vector4Compressed, 1153
 - operator !=, 1153
 - operator ==, 1153
 - W, 1155
 - wEncoded, 1154
 - X, 1155
 - xEncoded, 1154
 - Y, 1155
 - yEncoded, 1154
 - Z, 1155
 - zEncoded, 1154
- VerifyHash
 - IDataEncryption, 145
- VerifyRawNetworkUnwrap< T >
 - ReadWriteUtilsForWeaver, 851
- VerifyRawNetworkWrap< T >
 - ReadWriteUtilsForWeaver, 852
- Version
 - NetworkPrefabTable, 609
 - NetworkProjectConfig, 619
 - NetworkSceneInfo, 716
- Versioning, 1155
 - AssemblyFileVersion, 1156
 - GetCurrentVersion, 1157
 - InvalidVersion, 1157
 - ShortVersion, 1156
- Visibility
 - EditorButtonAttribute, 141
- W

- QuaternionCompressed, [838](#)
- Vector4Compressed, [1155](#)
- WaitForResult
 - INetworkAssetSource< T >, [235](#)
- Waiting
 - Fusion.Sockets, [70](#)
 - NetworkRunnerCallbackArgs.ConnectRequest, [699](#)
- WarnIfAttribute, [1157](#)
 - AsBox, [1158](#)
 - Message, [1158](#)
 - WarnIfAttribute, [1158](#)
- WarnIfNoPreserveAttribute
 - SerializableTypeAttribute, [899](#)
- WasPressed
 - NetworkButtons, [466](#)
- WasPressed< T >
 - NetworkButtons, [466](#)
- WasReleased
 - NetworkButtons, [467](#)
- WasReleased< T >
 - NetworkButtons, [467](#)
- WaveShape
 - NetConfigSimulationOscillator, [1024](#)
- WeaverGeneratedAttribute, [1159](#)
- wEncoded
 - QuaternionCompressed, [838](#)
 - Vector4Compressed, [1154](#)
- WindowSize
 - NetConfigNotify, [1020](#)
- WithLossNotifySequences
 - NetConfigSimulation, [1022](#)
- WORD_COUNT
 - NetworkPhysicsInfo, [582](#)
 - NetworkSceneInfo, [716](#)
 - SimulationInputHeader, [941](#)
- WordCount
 - DefaultForPropertyAttribute, [123](#)
 - IExportedWordCount, [1159](#)
 - Native, [363](#)
 - NetworkBehaviourWeavedAttribute, [456](#)
 - NetworkedWeavedAttribute, [487](#)
 - NetworkInput, [504](#)
 - NetworkInputWeavedAttribute, [507](#)
 - NetworkObjectHeader, [556](#)
 - NetworkStructWeavedAttribute, [759](#)
- WordOffset
 - DefaultForPropertyAttribute, [123](#)
 - NetworkedWeavedAttribute, [487](#)
- WORDS
 - NetworkObjectHeader, [557](#)
 - NetworkTRSPData, [768](#)
 - TickRate.Resolved, [1113](#)
- WordsReadCount
 - FusionStatisticsSnapshot, [1084](#)
- WordsReadSize
 - FusionStatisticsSnapshot, [1084](#)
- WordsToHex
 - BinUtils, [115](#), [116](#)
- WordsWrittenCount
 - FusionStatisticsSnapshot, [1085](#)
- WordsWrittenSize
 - FusionStatisticsSnapshot, [1085](#)
- WorldSnapshotSize
 - EngineProfiler, [153](#)
- WorldSnapshotSizeCallback
 - EngineProfiler, [156](#)
- Wrap
 - SerializableDictionary< TKey, TValue >, [893](#)
- Write
 - IElementReaderWriter< T >, [230](#)
 - IUnitySurrogate, [256](#)
 - NetBitBuffer, [989](#)
 - NetworkId, [499](#)
 - PlayerRef, [776](#)
 - RpcHeader, [866](#)
 - UnityArraySurrogate< T, ReaderWriter >, [258](#)
 - UnityDictionarySurrogate< TKeyType, TKeyReaderWriter, TValueType, TValueReaderWriter >, [261](#)
 - UnityLinkedListSurrogate< T, ReaderWriter >, [263](#)
 - UnitySurrogateBase, [265](#)
 - UnityValueSurrogate< T, TReaderWriter >, [266](#)
- Write< T >
 - PlayerRef, [776](#)
- WriteBool
 - BitStream, [812](#)
- WriteBoolean
 - BitStream, [813](#)
 - INetBitWriteStream, [956](#)
 - NetBitBuffer, [989](#)
 - NetBitBufferNull, [1003](#)
 - ReadWriteUtilsForWeaver, [852](#)
- WriteByte
 - BitStream, [813](#)
 - NetBitBuffer, [989](#)
 - NetBitBufferNull, [1003](#)
- WriteByteArray
 - BitStream, [814](#)
- WriteByteArrayLengthPrefixed
 - BitStream, [815](#)
- WriteByteAt
 - BitStream, [815](#)
- WriteBytesAligned
 - INetBitWriteStream, [956](#)
 - NetBitBuffer, [990](#)
 - NetBitBufferNull, [1004](#)
- WriteChar
 - BitStream, [817](#)
- WriteDouble
 - BitStream, [817](#)
 - NetBitBuffer, [990](#)
- WriteFloat
 - BitStream, [817](#)
 - ReadWriteUtils, [844](#)
- WriteGuid

- BitStream, [817](#)
 - WriteInt
 - BitStream, [818](#)
 - SimulationMessage, [948](#)
 - WriteInt16
 - NetBitBuffer, [991](#)
 - WriteInt32
 - INetBitWriteStream, [956](#)
 - NetBitBuffer, [991](#)
 - NetBitBufferNull, [1004](#)
 - WriteInt32AtOffset
 - NetBitBuffer, [991](#)
 - WriteInt32VarLength
 - INetBitWriteStream, [957](#)
 - NetBitBuffer, [992](#)
 - NetBitBufferNull, [1004](#), [1005](#)
 - WriteInt64
 - NetBitBuffer, [992](#)
 - WriteInt64VarLength
 - NetBitBuffer, [993](#)
 - WriteInt_Shifted
 - BitStream, [818](#)
 - WriteLengthPrefixedUTF8
 - Native, [364](#)
 - WriteLong
 - BitStream, [819](#)
 - WriteNetworkedObjectRef
 - SimulationMessage, [948](#)
 - WriteQuaternion
 - ReadWriteUtils, [845](#)
 - Writer
 - NetBitBufferSerializer, [1010](#)
 - WriteSByte
 - BitStream, [819](#)
 - WriteShort
 - BitStream, [820](#)
 - WriteSingle
 - NetBitBuffer, [993](#)
 - WriteSlow
 - NetBitBuffer, [993](#)
 - WriteString
 - BitStream, [820](#)
 - NetBitBuffer, [994](#)
 - WriteStringUtf32NoHash
 - ReadWriteUtilsForWeaver, [853](#)
 - WriteStringUtf32WithHash
 - ReadWriteUtilsForWeaver, [853](#)
 - WriteStringUtf8NoHash
 - ReadWriteUtilsForWeaver, [853](#)
 - WriteUInt
 - BitStream, [821](#)
 - WriteUInt16
 - NetBitBuffer, [994](#)
 - WriteUInt32
 - NetBitBuffer, [994](#)
 - WriteUInt32VarLength
 - NetBitBuffer, [995](#)
 - NetBitBufferNull, [1005](#)
 - WriteUInt64
 - NetBitBuffer, [995](#)
 - WriteUInt64AtOffset
 - NetBitBuffer, [996](#)
 - WriteUInt64VarLength
 - INetBitWriteStream, [957](#)
 - NetBitBuffer, [996](#)
 - NetBitBufferNull, [1006](#)
 - WriteULong
 - BitStream, [821](#), [822](#)
 - WriteUShort
 - BitStream, [822](#)
 - WriteVector2
 - ReadWriteUtils, [845](#)
 - WriteVector3
 - ReadWriteUtils, [845](#)
 - SimulationMessage, [949](#)
 - WriteVector4
 - ReadWriteUtils, [846](#)
 - Writing
 - BitStream, [824](#)
 - NetBitBufferSerializer, [1011](#)
- X
- QuaternionCompressed, [838](#)
 - Vector2Compressed, [1144](#)
 - Vector3Compressed, [1150](#)
 - Vector4Compressed, [1155](#)
- x
- Simulation.AreaOfInterest, [924](#)
- xEncoded
- QuaternionCompressed, [838](#)
 - Vector2Compressed, [1144](#)
 - Vector3Compressed, [1149](#)
 - Vector4Compressed, [1154](#)
- Y
- QuaternionCompressed, [839](#)
 - Vector2Compressed, [1144](#)
 - Vector3Compressed, [1150](#)
 - Vector4Compressed, [1155](#)
- yEncoded
- QuaternionCompressed, [838](#)
 - Vector2Compressed, [1144](#)
 - Vector3Compressed, [1149](#)
 - Vector4Compressed, [1154](#)
- Z
- QuaternionCompressed, [839](#)
 - Vector3Compressed, [1150](#)
 - Vector4Compressed, [1155](#)
- zEncoded
- QuaternionCompressed, [838](#)
 - Vector3Compressed, [1149](#)
 - Vector4Compressed, [1154](#)
- ZigZagDecode
- Maths, [331](#), [332](#)
- ZigZagEncode
- Maths, [332](#)