

Photon Fusion

2.0.0

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	29
5.1 Fusion Namespace Reference	29
5.1.1 Enumeration Type Documentation	40
5.1.1.1 CompareOperator	41
5.1.1.2 ConnectionType	41
5.1.1.3 GameMode	41
5.1.1.4 HitboxTypes	42
5.1.1.5 HitOptions	42
5.1.1.6 NetworkObjectAcquireResult	42
5.1.1.7 NetworkObjectFlags	43
5.1.1.8 NetworkObjectHeaderFlags	43
5.1.1.9 NetworkPrefabTableGetPrefabResult	44
5.1.1.10 NetworkSceneInfoChangeSource	44
5.1.1.11 NetworkSceneInfoDefaultFlags	44
5.1.1.12 NetworkSpawnFlags	44
5.1.1.13 NetworkSpawnStatus	45
5.1.1.14 NetworkTypeIdKind	45
5.1.1.15 PageSizes	45
5.1.1.16 PriorityLevel	46
5.1.1.17 RenderSource	46
5.1.1.18 RenderTimeframe	46
5.1.1.19 RpcChannel	47
5.1.1.20 RpcHostMode	47
5.1.1.21 RpcLocallInvokeResult	47
5.1.1.22 RpcSendCullResult	48
5.1.1.23 RpcSendMessageResult	48

5.1.1.24 RpcSources	49
5.1.1.25 RpcTargets	49
5.1.1.26 RpcTargetStatus	49
5.1.1.27 ScriptHeaderBackColor	50
5.1.1.28 ScriptHeaderIcon	50
5.1.1.29 SessionLobby	50
5.1.1.30 ShutdownReason	50
5.1.1.31 SimulationModes	51
5.1.1.32 SimulationStages	51
5.1.1.33 Topologies	52
5.1.1.34 Units	52
5.1.1.35 UnityPlayerLoopSystemAddMode	53
5.1.2 Function Documentation	53
5.1.2.1 NetworkObjectSpawnDelegate()	53
5.1.2.2 RpcInvokeDelegate()	53
5.1.2.3 RpcStaticInvokeDelegate()	53
5.2 Fusion.Analyzer Namespace Reference	54
5.3 Fusion.Async Namespace Reference	54
5.4 Fusion.Encryption Namespace Reference	54
5.5 Fusion.Internal Namespace Reference	54
5.6 Fusion.LagCompensation Namespace Reference	55
5.6.1 Enumeration Type Documentation	56
5.6.1.1 HitType	56
5.6.2 Function Documentation	56
5.6.2.1 PreProcessingDelegate()	56
5.7 Fusion.Protocol Namespace Reference	57
5.7.1 Enumeration Type Documentation	57
5.7.1.1 DisconnectReason	57
5.8 Fusion.Runtime Namespace Reference	57
5.9 Fusion.Runtime.Unity Namespace Reference	57
5.10 Fusion.Sockets Namespace Reference	57
5.10.1 Enumeration Type Documentation	58
5.10.1.1 NetCommands	58
5.10.1.2 NetConnectFailedReason	58
5.10.1.3 NetDisconnectReason	59
5.10.1.4 NetPacketType	59
5.11 Fusion.Sockets.Stun Namespace Reference	59
5.11.1 Enumeration Type Documentation	59
5.11.1.1 NATType	59
5.12 Fusion.Statistics Namespace Reference	60
5.13 UnityEngine Namespace Reference	60
5.14 UnityEngine.SceneManagement Namespace Reference	60

5.15 UnityEngine.Scripting Namespace Reference	60
5.16 UnityEngine.Serialization Namespace Reference	60
6 Class Documentation	61
6.1 _128 Struct Reference	61
6.1.1 Detailed Description	61
6.1.2 Member Data Documentation	61
6.1.2.1 Data	61
6.1.2.2 SIZE	62
6.2 _16 Struct Reference	62
6.2.1 Detailed Description	62
6.2.2 Member Data Documentation	62
6.2.2.1 Data	62
6.2.2.2 SIZE	62
6.3 _2 Struct Reference	63
6.3.1 Detailed Description	63
6.3.2 Member Data Documentation	63
6.3.2.1 Data	63
6.3.2.2 SIZE	63
6.4 _256 Struct Reference	63
6.4.1 Detailed Description	64
6.4.2 Member Data Documentation	64
6.4.2.1 Data	64
6.4.2.2 SIZE	64
6.5 _32 Struct Reference	64
6.5.1 Detailed Description	65
6.5.2 Member Data Documentation	65
6.5.2.1 Data	65
6.5.2.2 SIZE	65
6.6 _4 Struct Reference	65
6.6.1 Detailed Description	66
6.6.2 Member Data Documentation	66
6.6.2.1 Data	66
6.6.2.2 SIZE	66
6.7 _512 Struct Reference	66
6.7.1 Detailed Description	66
6.7.2 Member Data Documentation	67
6.7.2.1 Data	67
6.7.2.2 SIZE	67
6.8 _64 Struct Reference	67
6.8.1 Detailed Description	67
6.8.2 Member Data Documentation	67

6.8.2.1 Data	68
6.8.2.2 SIZE	68
6.9 _8 Struct Reference	68
6.9.1 Detailed Description	68
6.9.2 Member Data Documentation	68
6.9.2.1 Data	68
6.9.2.2 SIZE	69
6.10 Allocator Struct Reference	69
6.10.1 Detailed Description	70
6.10.2 Member Data Documentation	70
6.10.2.1 BUCKET_COUNT	70
6.10.2.2 BUCKET_INVALID	70
6.10.2.3 HEAP_ALIGNMENT	70
6.10.2.4 REPLICATE_WORD_ALIGN	71
6.10.2.5 REPLICATE_WORD_SHIFT	71
6.10.2.6 REPLICATE_WORD_SIZE	71
6.11 Allocator.Config Struct Reference	71
6.11.1 Detailed Description	72
6.11.2 Constructor & Destructor Documentation	72
6.11.2.1 Config()	72
6.11.3 Member Function Documentation	72
6.11.3.1 Equals() [1/2]	73
6.11.3.2 Equals() [2/2]	73
6.11.3.3 GetHashCode()	73
6.11.3.4 ToString()	74
6.11.4 Member Data Documentation	74
6.11.4.1 BlockCount	74
6.11.4.2 BlockShift	74
6.11.4.3 DEFAULT_BLOCK_COUNT	74
6.11.4.4 DEFAULT_BLOCK_SHIFT	74
6.11.4.5 GlobalsSize	74
6.11.4.6 SIZE	75
6.11.5 Property Documentation	75
6.11.5.1 BlockByteSize	75
6.11.5.2 BlockWordCount	75
6.11.5.3 HeapSizeAllocated	75
6.11.5.4 HeapSizeUsable	75
6.12 Angle Struct Reference	75
6.12.1 Detailed Description	77
6.12.2 Member Function Documentation	77
6.12.2.1 Clamp() [1/2]	77
6.12.2.2 Clamp() [2/2]	77

6.12.2.3 Equals() [1/2]	77
6.12.2.4 Equals() [2/2]	78
6.12.2.5 GetHashCode()	78
6.12.2.6 Lerp()	78
6.12.2.7 Max()	79
6.12.2.8 Min()	79
6.12.2.9 operator Angle() [1/3]	79
6.12.2.10 operator Angle() [2/3]	79
6.12.2.11 operator Angle() [3/3]	80
6.12.2.12 operator double()	80
6.12.2.13 operator float()	80
6.12.2.14 operator"!="()	81
6.12.2.15 operator+()	81
6.12.2.16 operator-()	82
6.12.2.17 operator<()	82
6.12.2.18 operator<=()	82
6.12.2.19 operator==()	83
6.12.2.20 operator>()	83
6.12.2.21 operator>=()	84
6.12.2.22 ToString()	84
6.12.3 Member Data Documentation	84
6.12.3.1 SIZE	84
6.13 AssetObject Class Reference	84
6.13.1 Detailed Description	85
6.14 TaskManager Class Reference	85
6.14.1 Detailed Description	85
6.14.2 Member Function Documentation	85
6.14.2.1 ContinueWhenAll()	85
6.14.2.2 Run()	86
6.14.2.3 Service()	86
6.14.2.4 Setup()	87
6.15 AuthorityMasks Class Reference	87
6.15.1 Detailed Description	87
6.15.2 Member Data Documentation	87
6.15.2.1 ALL	87
6.15.2.2 INPUT	87
6.15.2.3 NONE	88
6.15.2.4 PROXY	88
6.15.2.5 STATE	88
6.16 Behaviour Class Reference	88
6.16.1 Detailed Description	89
6.16.2 Member Function Documentation	89

6.16.2.1 AddBehaviour< T >()	89
6.16.2.2 DestroyBehaviour()	89
6.16.2.3 GetBehaviour< T >()	89
6.16.2.4 TryGetBehaviour< T >()	90
6.17 CapacityAttribute Class Reference	90
6.17.1 Detailed Description	90
6.17.2 Constructor & Destructor Documentation	90
6.17.2.1 CapacityAttribute()	90
6.17.3 Property Documentation	91
6.17.3.1 Length	91
6.18 DefaultForPropertyAttribute Class Reference	91
6.18.1 Detailed Description	91
6.18.2 Constructor & Destructor Documentation	91
6.18.2.1 DefaultForPropertyAttribute()	91
6.18.3 Property Documentation	92
6.18.3.1 PropertyName	92
6.18.3.2 WordCount	92
6.18.3.3 WordOffset	92
6.19 DisplayAsEnumAttribute Class Reference	92
6.19.1 Detailed Description	93
6.20 DolfAttributeBase Class Reference	93
6.20.1 Detailed Description	93
6.21 DrawlfAttribute Class Reference	93
6.21.1 Detailed Description	94
6.21.2 Constructor & Destructor Documentation	94
6.21.2.1 DrawlfAttribute() [1/2]	94
6.21.2.2 DrawlfAttribute() [2/2]	95
6.21.3 Member Data Documentation	95
6.21.3.1 Mode	95
6.22 DynamicHeap Struct Reference	95
6.22.1 Detailed Description	97
6.22.2 Member Function Documentation	97
6.22.2.1 CollectGarbage()	97
6.22.2.2 CollectGarbageDelegate()	98
6.22.2.3 Free()	98
6.22.2.4 SetForcedAlive< T >()	98
6.22.3 Member Data Documentation	99
6.22.3.1 _debruijnTable	99
6.23 DynamicHeap.Ignore Class Reference	99
6.23.1 Detailed Description	99
6.24 DynamicHeapInstance Class Reference	99
6.24.1 Detailed Description	100

6.24.2 Constructor & Destructor Documentation	100
6.24.2.1 DynamicHeapInstance()	100
6.24.3 Member Function Documentation	100
6.24.3.1 Allocate()	100
6.24.3.2 AllocateArray< T >()	100
6.24.3.3 AllocateArrayPointers< T >()	101
6.24.3.4 AllocateTracked< T >()	101
6.24.3.5 AllocateTrackedArray< T >()	102
6.24.3.6 AllocateTrackedArrayPointers< T >()	103
6.24.3.7 Free()	104
6.25 DataEncryptor Class Reference	104
6.25.1 Detailed Description	105
6.25.2 Member Function Documentation	105
6.25.2.1 EncryptData()	105
6.26 IDataEncryption Interface Reference	105
6.26.1 Detailed Description	106
6.26.2 Member Function Documentation	106
6.26.2.1 ComputeHash()	106
6.26.2.2 DecryptData()	107
6.26.2.3 EncryptData()	107
6.26.2.4 GenerateKey()	107
6.26.2.5 Setup()	108
6.26.2.6 VerifyHash()	108
6.27 FieldsMask< T > Class Template Reference	108
6.27.1 Detailed Description	109
6.27.2 Constructor & Destructor Documentation	109
6.27.2.1 FieldsMask() [1/5]	109
6.27.2.2 FieldsMask() [2/5]	110
6.27.2.3 FieldsMask() [3/5]	110
6.27.2.4 FieldsMask() [4/5]	110
6.27.2.5 FieldsMask() [5/5]	110
6.27.3 Member Function Documentation	110
6.27.3.1 operator Mask256()	110
6.28 FixedArray< T > Class Template Reference	111
6.28.1 Detailed Description	112
6.28.2 Constructor & Destructor Documentation	112
6.28.2.1 FixedArray()	112
6.28.3 Member Function Documentation	112
6.28.3.1 Clear()	113
6.28.3.2 CopyFrom() [1/2]	113
6.28.3.3 CopyFrom() [2/2]	113
6.28.3.4 CopyTo() [1/2]	114

6.28.3.5 CopyTo() [2/2]	114
6.28.3.6 Create< T >()	114
6.28.3.7 Create< TActual, TAdapted >()	115
6.28.3.8 CreateFromFieldSequence< T >()	115
6.28.3.9 GetEnumerator()	116
6.28.3.10 IndexOf< T >()	116
6.28.3.11 ToArray()	116
6.28.3.12 ToStringString()	117
6.28.3.13 ToString()	117
6.28.4 Property Documentation	117
6.28.4.1 Length	117
6.28.4.2 this[int index]	117
6.29 FixedArray< T >.Enumerator Struct Reference	117
6.29.1 Detailed Description	118
6.29.2 Constructor & Destructor Documentation	118
6.29.2.1 Enumerator()	118
6.29.3 Member Function Documentation	118
6.29.3.1 Dispose()	118
6.29.3.2 MoveNext()	119
6.29.3.3 Reset()	119
6.29.4 Property Documentation	119
6.29.4.1 Current	119
6.30 FixedBufferPropertyAttribute Class Reference	119
6.30.1 Detailed Description	120
6.30.2 Constructor & Destructor Documentation	120
6.30.2.1 FixedBufferPropertyAttribute()	120
6.30.3 Property Documentation	120
6.30.3.1 Capacity	120
6.30.3.2 SurrogateType	120
6.30.3.3 Type	121
6.31 FixedStorage Class Reference	121
6.31.1 Detailed Description	121
6.31.2 Member Function Documentation	121
6.31.2.1 GetWordCount< T >()	121
6.32 FloatCompressed Struct Reference	122
6.32.1 Detailed Description	122
6.32.2 Member Function Documentation	122
6.32.2.1 Equals() [1/2]	122
6.32.2.2 Equals() [2/2]	123
6.32.2.3 GetHashCode()	123
6.32.2.4 operator float()	123
6.32.2.5 operator FloatCompressed()	124

6.32.2.6 operator"!="	124
6.32.2.7 operator==()	125
6.32.3 Member Data Documentation	125
6.32.3.1 valueEncoded	125
6.33 FloatUtils Class Reference	125
6.33.1 Detailed Description	126
6.33.2 Member Function Documentation	126
6.33.2.1 Compress()	126
6.33.2.2 Decompress()	126
6.33.3 Member Data Documentation	126
6.33.3.1 DEFAULT_ACCURACY	127
6.34 HeapConfiguration Class Reference	127
6.34.1 Detailed Description	127
6.34.2 Member Function Documentation	127
6.34.2.1 Init()	127
6.34.2.2 ToString()	128
6.34.3 Member Data Documentation	128
6.34.3.1 GlobalsSize	128
6.34.3.2 PageCount	128
6.34.3.3 PageShift	128
6.35 Hitbox Class Reference	128
6.35.1 Detailed Description	129
6.35.2 Member Function Documentation	129
6.35.2.1 DrawGizmos()	130
6.35.2.2 OnDrawGizmos()	130
6.35.2.3 SetLayer()	130
6.35.3 Member Data Documentation	130
6.35.3.1 BoxExtents	130
6.35.3.2 CapsuleExtents	130
6.35.3.3 CapsuleRadius	131
6.35.3.4 GizmosColor	131
6.35.3.5 Offset	131
6.35.3.6 Root	131
6.35.3.7 SphereRadius	131
6.35.3.8 Type	131
6.35.4 Property Documentation	132
6.35.4.1 ColliderIndex	132
6.35.4.2 HitboxActive	132
6.35.4.3 HitboxIndex	132
6.35.4.4 Position	132
6.36 HitboxManager Class Reference	132
6.36.1 Detailed Description	134

6.36.2 Member Function Documentation	134
6.36.2.1 GetPlayerTickAndAlpha()	135
6.36.2.2 OverlapBox() [1/3]	135
6.36.2.3 OverlapBox() [2/3]	135
6.36.2.4 OverlapBox() [3/3]	136
6.36.2.5 OverlapSphere() [1/3]	137
6.36.2.6 OverlapSphere() [2/3]	138
6.36.2.7 OverlapSphere() [3/3]	139
6.36.2.8 PositionRotation() [1/2]	139
6.36.2.9 PositionRotation() [2/2]	140
6.36.2.10 Raycast() [1/3]	140
6.36.2.11 Raycast() [2/3]	141
6.36.2.12 Raycast() [3/3]	142
6.36.2.13 RaycastAll() [1/3]	143
6.36.2.14 RaycastAll() [2/3]	143
6.36.2.15 RaycastAll() [3/3]	144
6.36.3 Member Data Documentation	145
6.36.3.1 BVHDepth	145
6.36.3.2 BVHNodes	145
6.36.3.3 DrawInfo	145
6.36.3.4 TotalHitboxes	146
6.37 HitboxRoot Class Reference	146
6.37.1 Detailed Description	147
6.37.2 Member Enumeration Documentation	147
6.37.2.1 ConfigFlags	147
6.37.3 Member Function Documentation	148
6.37.3.1 DrawGizmos()	148
6.37.3.2 InitHitboxes()	148
6.37.3.3 IsHitboxActive()	148
6.37.3.4 OnDrawGizmos()	148
6.37.3.5 SetHitboxActive()	149
6.37.3.6 SetMinBoundingRadius()	149
6.37.4 Member Data Documentation	149
6.37.4.1 BroadRadius	149
6.37.4.2 Config	150
6.37.4.3 GizmosColor	150
6.37.4.4 Hitboxes	150
6.37.4.5 MAX_HITBOXES	150
6.37.4.6 Offset	150
6.37.5 Property Documentation	150
6.37.5.1 HitboxRootActive	151
6.37.5.2 InInterest	151

6.37.5.3 Manager	151
6.38 HostMigrationConfig Class Reference	151
6.38.1 Detailed Description	151
6.38.2 Member Data Documentation	151
6.38.2.1 EnableAutoUpdate	152
6.38.2.2 UpdateDelay	152
6.39 HostMigrationToken Class Reference	152
6.39.1 Detailed Description	152
6.39.2 Property Documentation	152
6.39.2.1 GameMode	152
6.40 IAfterAllTicks Interface Reference	152
6.40.1 Detailed Description	153
6.40.2 Member Function Documentation	153
6.40.2.1 AfterAllTicks()	153
6.41 IAfterClientPredictionReset Interface Reference	153
6.41.1 Detailed Description	153
6.41.2 Member Function Documentation	154
6.41.2.1 AfterClientPredictionReset()	154
6.42 IAfterHostMigration Interface Reference	154
6.42.1 Detailed Description	154
6.42.2 Member Function Documentation	154
6.42.2.1 AfterHostMigration()	154
6.43 IAfterRender Interface Reference	154
6.43.1 Detailed Description	155
6.43.2 Member Function Documentation	155
6.43.2.1 AfterRender()	155
6.44 IAfterSpawned Interface Reference	155
6.44.1 Detailed Description	155
6.44.2 Member Function Documentation	155
6.44.2.1 AfterSpawned()	155
6.45 IAfterTick Interface Reference	155
6.45.1 Detailed Description	156
6.45.2 Member Function Documentation	156
6.45.2.1 AfterTick()	156
6.46 IAfterUpdate Interface Reference	156
6.46.1 Detailed Description	156
6.46.2 Member Function Documentation	156
6.46.2.1 AfterUpdate()	156
6.47 IAsyncOperation Interface Reference	157
6.47.1 Detailed Description	157
6.47.2 Property Documentation	157
6.47.2.1 Error	157

6.47.2.2 IsDone	157
6.47.3 Event Documentation	157
6.47.3.1 Completed	158
6.48 IBeforeAllTicks Interface Reference	158
6.48.1 Detailed Description	158
6.48.2 Member Function Documentation	158
6.48.2.1 BeforeAllTicks()	158
6.49 IBeforeClientPredictionReset Interface Reference	159
6.49.1 Detailed Description	159
6.49.2 Member Function Documentation	159
6.49.2.1 BeforeClientPredictionReset()	159
6.50 IBeforeCopyPreviousState Interface Reference	159
6.50.1 Detailed Description	159
6.50.2 Member Function Documentation	160
6.50.2.1 BeforeCopyPreviousState()	160
6.51 IBeforeHitboxRegistration Interface Reference	160
6.51.1 Detailed Description	160
6.51.2 Member Function Documentation	160
6.51.2.1 BeforeHitboxRegistration()	160
6.52 IBeforeSimulation Interface Reference	160
6.52.1 Detailed Description	161
6.52.2 Member Function Documentation	161
6.52.2.1 BeforeSimulation()	161
6.53 IBeforeTick Interface Reference	161
6.53.1 Detailed Description	161
6.53.2 Member Function Documentation	162
6.53.2.1 BeforeTick()	162
6.54 IBeforeUpdate Interface Reference	162
6.54.1 Detailed Description	162
6.54.2 Member Function Documentation	162
6.54.2.1 BeforeUpdate()	162
6.55 ICORoutine Interface Reference	162
6.55.1 Detailed Description	163
6.56 IDespawned Interface Reference	163
6.56.1 Detailed Description	163
6.56.2 Member Function Documentation	163
6.56.2.1 Despawned()	163
6.57 IELEMENTREADERWRITER< T > Interface Template Reference	163
6.57.1 Detailed Description	164
6.57.2 Member Function Documentation	164
6.57.2.1 GetElementHashCode()	164
6.57.2.2 GetElementWordCount()	164

6.57.2.3 Read()	165
6.57.2.4 ReadRef()	165
6.57.2.5 Write()	165
6.58 IFixedStorage Interface Reference	167
6.58.1 Detailed Description	167
6.59 IInputAuthorityGained Interface Reference	167
6.59.1 Detailed Description	167
6.59.2 Member Function Documentation	167
6.59.2.1 InputAuthorityGained()	167
6.60 IInputAuthorityLost Interface Reference	168
6.60.1 Detailed Description	168
6.60.2 Member Function Documentation	168
6.60.2.1 InputAuthorityLost()	168
6.61 IIInterestEnter Interface Reference	168
6.61.1 Detailed Description	168
6.61.2 Member Function Documentation	168
6.61.2.1 InterestEnter()	168
6.62 IIInterestExit Interface Reference	169
6.62.1 Detailed Description	169
6.62.2 Member Function Documentation	169
6.62.2.1 InterestExit()	169
6.63 ILocalPrefabCreated Interface Reference	169
6.63.1 Detailed Description	170
6.63.2 Member Function Documentation	170
6.63.2.1 LocalPrefabCreated()	170
6.64 INetworkArray Interface Reference	170
6.64.1 Detailed Description	170
6.64.2 Property Documentation	170
6.64.2.1 this[int index]	170
6.65 INetworkAssetSource< T > Interface Template Reference	171
6.65.1 Detailed Description	171
6.65.2 Member Function Documentation	171
6.65.2.1 Acquire()	171
6.65.2.2 Release()	172
6.65.2.3 WaitForResult()	172
6.65.3 Property Documentation	172
6.65.3.1 Description	172
6.65.3.2 IsCompleted	172
6.66 INetworkDictionary Interface Reference	172
6.66.1 Detailed Description	173
6.66.2 Member Function Documentation	173
6.66.2.1 Add()	173

6.67 INetworkInput Interface Reference	173
6.67.1 Detailed Description	173
6.68 INetworkLinkedList Interface Reference	173
6.68.1 Detailed Description	174
6.68.2 Member Function Documentation	174
6.68.2.1 Add()	174
6.69 INetworkObjectInitializer Interface Reference	174
6.69.1 Detailed Description	174
6.69.2 Member Function Documentation	174
6.69.2.1 InitializeNetworkState()	175
6.70 INetworkObjectProvider Interface Reference	175
6.70.1 Detailed Description	175
6.70.2 Member Function Documentation	175
6.70.2.1 AcquirePrefabInstance()	175
6.70.2.2 ReleaseInstance()	176
6.71 INetworkPrefabSource Interface Reference	176
6.71.1 Detailed Description	176
6.71.2 Property Documentation	177
6.71.2.1 AssetGuid	177
6.72 INetworkRunnerCallbacks Interface Reference	177
6.72.1 Detailed Description	178
6.72.2 Member Function Documentation	178
6.72.2.1 OnConnectedToServer()	178
6.72.2.2 OnConnectFailed()	178
6.72.2.3 OnConnectRequest()	178
6.72.2.4 OnCustomAuthenticationResponse()	179
6.72.2.5 OnDisconnectedFromServer()	179
6.72.2.6 OnHostMigration()	179
6.72.2.7 OnInput()	180
6.72.2.8 OnInputMissing()	180
6.72.2.9 OnObjectEnterAOI()	180
6.72.2.10 OnObjectExitAOI()	180
6.72.2.11 OnPlayerJoined()	181
6.72.2.12 OnPlayerLeft()	181
6.72.2.13 OnReliableDataProgress()	181
6.72.2.14 OnReliableDataReceived()	182
6.72.2.15 OnSceneLoadDone()	182
6.72.2.16 OnSceneLoadStart()	182
6.72.2.17 OnSessionListUpdated()	183
6.72.2.18 OnShutdown()	183
6.72.2.19 OnUserSimulationMessage()	183
6.73 INetworkRunnerUpdater Interface Reference	183

6.73.1 Detailed Description	184
6.73.2 Member Function Documentation	184
6.73.2.1 Initialize()	184
6.73.2.2 Shutdown()	184
6.74 INetworkSceneManager Interface Reference	185
6.74.1 Detailed Description	185
6.74.2 Member Function Documentation	185
6.74.2.1 GetSceneRef() [1/2]	186
6.74.2.2 GetSceneRef() [2/2]	186
6.74.2.3 Initialize()	186
6.74.2.4 IsRunnerScene()	186
6.74.2.5 LoadScene()	186
6.74.2.6 MakeDontDestroyOnLoad()	187
6.74.2.7 MoveGameObjectToScene()	187
6.74.2.8 OnSceneInfoChanged()	187
6.74.2.9 Shutdown()	187
6.74.2.10 TryGetPhysicsScene2D()	188
6.74.2.11 TryGetPhysicsScene3D()	188
6.74.2.12 UnloadScene()	188
6.74.3 Property Documentation	188
6.74.3.1 IsBusy	188
6.74.3.2 MainRunnerScene	189
6.75 INetworkStruct Interface Reference	189
6.75.1 Detailed Description	189
6.76 INetworkTRSPTeleport Interface Reference	189
6.76.1 Detailed Description	189
6.76.2 Member Function Documentation	189
6.76.2.1 Teleport()	190
6.77 IUnitySurrogate Interface Reference	190
6.77.1 Detailed Description	190
6.77.2 Member Function Documentation	190
6.77.2.1 Read()	190
6.77.2.2 Write()	191
6.78 IUnityValueSurrogate< T > Interface Template Reference	191
6.78.1 Detailed Description	191
6.78.2 Property Documentation	192
6.78.2.1 DataProperty	192
6.79 UnityArraySurrogate< T, ReaderWriter > Class Template Reference	192
6.79.1 Detailed Description	192
6.79.2 Member Function Documentation	193
6.79.2.1 Init()	193
6.79.2.2 Read()	193

6.79.2.3 Write()	193
6.79.3 Property Documentation	194
6.79.3.1 DataProperty	194
6.80 UnityDictionarySurrogate< TKeyType, TKeyReaderWriter, TValueType, TValueReaderWriter > Class Template Reference	194
6.80.1 Detailed Description	194
6.80.2 Member Function Documentation	195
6.80.2.1 Init()	195
6.80.2.2 Read()	195
6.80.2.3 Write()	196
6.80.3 Property Documentation	196
6.80.3.1 DataProperty	196
6.81 UnityLinkedListSurrogate< T, ReaderWriter > Class Template Reference	196
6.81.1 Detailed Description	196
6.81.2 Member Function Documentation	197
6.81.2.1 Init()	197
6.81.2.2 Read()	197
6.81.2.3 Write()	198
6.81.3 Property Documentation	198
6.81.3.1 DataProperty	198
6.82 UnitySurrogateBase Class Reference	198
6.82.1 Detailed Description	199
6.82.2 Member Function Documentation	199
6.82.2.1 Init()	199
6.82.2.2 Read()	199
6.82.2.3 Write()	199
6.83 UnityValueSurrogate< T, TReaderWriter > Class Template Reference	200
6.83.1 Detailed Description	200
6.83.2 Member Function Documentation	201
6.83.2.1 Init()	201
6.83.2.2 Read()	201
6.83.2.3 Write()	201
6.83.3 Property Documentation	202
6.83.3.1 DataProperty	202
6.84 InterpolatedErrorCorrectionSettings Class Reference	202
6.84.1 Detailed Description	202
6.84.2 Member Data Documentation	202
6.84.2.1 MaxRate	203
6.84.2.2 MinRate	203
6.84.2.3 PosBlendEnd	203
6.84.2.4 PosBlendStart	203
6.84.2.5 PosMinCorrection	204

6.84.2.6 PosTeleportDistance	204
6.84.2.7 RotBlendEnd	204
6.84.2.8 RotBlendStart	204
6.84.2.9 RotTeleportRadians	205
6.85 IPlayerJoined Interface Reference	205
6.85.1 Detailed Description	205
6.85.2 Member Function Documentation	205
6.85.2.1 PlayerJoined()	205
6.86 IPlayerLeft Interface Reference	205
6.86.1 Detailed Description	206
6.86.2 Member Function Documentation	206
6.86.2.1 PlayerLeft()	206
6.87 IRemotePrefabCreated Interface Reference	206
6.87.1 Detailed Description	206
6.87.2 Member Function Documentation	206
6.87.2.1 RemotePrefabCreated()	207
6.88 ISceneLoadDone Interface Reference	207
6.88.1 Detailed Description	207
6.88.2 Member Function Documentation	207
6.88.2.1 SceneLoadDone()	207
6.89 ISceneLoadStart Interface Reference	207
6.89.1 Detailed Description	208
6.89.2 Member Function Documentation	208
6.89.2.1 SceneLoadStart()	208
6.90 ISimulationEnter Interface Reference	208
6.90.1 Detailed Description	208
6.90.2 Member Function Documentation	208
6.90.2.1 SimulationEnter()	208
6.91 ISimulationExit Interface Reference	209
6.91.1 Detailed Description	209
6.91.2 Member Function Documentation	209
6.91.2.1 SimulationExit()	209
6.92 ISpawned Interface Reference	209
6.92.1 Detailed Description	209
6.92.2 Member Function Documentation	210
6.92.2.1 Spawned()	210
6.93 IStateAuthorityChanged Interface Reference	210
6.93.1 Detailed Description	210
6.93.2 Member Function Documentation	210
6.93.2.1 StateAuthorityChanged()	210
6.94 LagCompensatedHit Struct Reference	210
6.94.1 Detailed Description	211

6.94.2 Member Function Documentation	211
6.94.2.1 operator LagCompensatedHit() [1/2]	211
6.94.2.2 operator LagCompensatedHit() [2/2]	212
6.94.3 Member Data Documentation	212
6.94.3.1 Collider	212
6.94.3.2 Collider2D	212
6.94.3.3 Distance	213
6.94.3.4 GameObject	213
6.94.3.5 Hitbox	213
6.94.3.6 HitboxColliderPosition	213
6.94.3.7 HitboxColliderRotation	213
6.94.3.8 Normal	213
6.94.3.9 Point	214
6.94.3.10 Type	214
6.95 AABB Struct Reference	214
6.95.1 Detailed Description	214
6.95.2 Constructor & Destructor Documentation	214
6.95.2.1 AABB() [1/2]	214
6.95.2.2 AABB() [2/2]	215
6.95.3 Member Data Documentation	215
6.95.3.1 Center	215
6.95.3.2 Extents	215
6.95.3.3 Max	215
6.95.3.4 Min	216
6.96 BoxOverlapQuery Class Reference	216
6.96.1 Detailed Description	216
6.96.2 Constructor & Destructor Documentation	216
6.96.2.1 BoxOverlapQuery() [1/2]	216
6.96.2.2 BoxOverlapQuery() [2/2]	217
6.96.3 Member Function Documentation	217
6.96.3.1 Check()	217
6.96.4 Member Data Documentation	217
6.96.4.1 Center	218
6.96.4.2 Extents	218
6.96.4.3 Rotation	218
6.97 BoxOverlapQueryParams Struct Reference	218
6.97.1 Detailed Description	219
6.97.2 Constructor & Destructor Documentation	219
6.97.2.1 BoxOverlapQueryParams()	219
6.97.3 Member Data Documentation	219
6.97.3.1 Center	219
6.97.3.2 Extents	219

6.97.3.3 QueryParams	220
6.97.3.4 Rotation	220
6.97.3.5 StaticHitsCapacity	220
6.98 BVHDraw Class Reference	220
6.98.1 Detailed Description	220
6.98.2 Member Function Documentation	220
6.98.2.1 GetEnumerator()	220
6.99 BVHNodeDrawInfo Class Reference	221
6.99.1 Detailed Description	221
6.99.2 Property Documentation	221
6.99.2.1 Bounds	221
6.99.2.2 Depth	221
6.99.2.3 MaxDepth	221
6.100 ColliderDrawInfo Class Reference	222
6.100.1 Detailed Description	222
6.100.2 Property Documentation	222
6.100.2.1 BoxExtents	222
6.100.2.2 CapsuleBottomCenter	222
6.100.2.3 CapsuleExtents	223
6.100.2.4 CapsuleTopCenter	223
6.100.2.5 LocalToWorldMatrix	223
6.100.2.6 Offset	223
6.100.2.7 Radius	223
6.100.2.8 Type	223
6.101 HitboxColliderContainerDraw Class Reference	224
6.101.1 Detailed Description	224
6.101.2 Member Function Documentation	224
6.101.2.1 GetEnumerator()	224
6.102 LagCompensatedExt Class Reference	224
6.102.1 Detailed Description	224
6.102.2 Member Function Documentation	224
6.102.2.1 SortDistance()	224
6.102.2.2 SortReference()	225
6.103 LagCompensationDraw Class Reference	225
6.103.1 Detailed Description	226
6.103.2 Member Function Documentation	226
6.103.2.1 GizmosDrawWireCapsule()	226
6.103.3 Member Data Documentation	226
6.103.3.1 BVHDraw	226
6.103.3.2 SnapshotHistoryDraw	226
6.104 LagCompensationUtils.ContactData Struct Reference	226
6.104.1 Detailed Description	227

6.104.2 Member Data Documentation	227
6.104.2.1 Normal	227
6.104.2.2 Penetration	227
6.104.2.3 Point	227
6.105 PositionRotationQueryParams Struct Reference	227
6.105.1 Detailed Description	228
6.105.2 Constructor & Destructor Documentation	228
6.105.2.1 PositionRotationQueryParams()	228
6.105.3 Member Data Documentation	228
6.105.3.1 Hitbox	228
6.105.3.2 QueryParams	229
6.106 Query Class Reference	229
6.106.1 Detailed Description	229
6.106.2 Constructor & Destructor Documentation	230
6.106.2.1 Query()	230
6.106.3 Member Function Documentation	231
6.106.3.1 Check()	231
6.106.4 Member Data Documentation	231
6.106.4.1 Alpha	231
6.106.4.2 LayerMask	231
6.106.4.3 Options	232
6.106.4.4 Player	232
6.106.4.5 PreProcessingDelegate	232
6.106.4.6 Tick	232
6.106.4.7 TickTo	232
6.106.4.8 TriggerInteraction	232
6.106.4.9 UserArgs	233
6.107 queryParams Struct Reference	233
6.107.1 Detailed Description	233
6.107.2 Member Data Documentation	233
6.107.2.1 Alpha	233
6.107.2.2 LayerMask	234
6.107.2.3 Options	234
6.107.2.4 Player	234
6.107.2.5 PreProcessingDelegate	234
6.107.2.6 Tick	234
6.107.2.7 TickTo	234
6.107.2.8 TriggerInteraction	235
6.107.2.9 UserArgs	235
6.108 RaycastAllQuery Class Reference	235
6.108.1 Detailed Description	235
6.108.2 Constructor & Destructor Documentation	235

6.108.2.1 RaycastAllQuery() [1/2]	235
6.108.2.2 RaycastAllQuery() [2/2]	236
6.109 RaycastQuery Class Reference	236
6.109.1 Detailed Description	237
6.109.2 Constructor & Destructor Documentation	237
6.109.2.1 RaycastQuery()	237
6.109.3 Member Function Documentation	237
6.109.3.1 Check()	237
6.109.4 Member Data Documentation	238
6.109.4.1 Direction	238
6.109.4.2 Length	238
6.109.4.3 Origin	238
6.110 RaycastQueryParams Struct Reference	238
6.110.1 Detailed Description	239
6.110.2 Constructor & Destructor Documentation	239
6.110.2.1 RaycastQueryParams()	239
6.110.3 Member Data Documentation	239
6.110.3.1 Direction	239
6.110.3.2 Length	239
6.110.3.3 Origin	240
6.110.3.4 QueryParams	240
6.110.3.5 StaticHitsCapacity	240
6.111 SnapshotHistoryDraw Class Reference	240
6.111.1 Detailed Description	240
6.111.2 Member Function Documentation	240
6.111.2.1 GetEnumerator()	241
6.112 SphereOverlapQuery Class Reference	241
6.112.1 Detailed Description	241
6.112.2 Constructor & Destructor Documentation	241
6.112.2.1 SphereOverlapQuery() [1/2]	241
6.112.2.2 SphereOverlapQuery() [2/2]	242
6.112.3 Member Function Documentation	242
6.112.3.1 Check()	242
6.112.4 Member Data Documentation	242
6.112.4.1 Center	243
6.112.4.2 Radius	243
6.113 SphereOverlapQueryParams Struct Reference	243
6.113.1 Detailed Description	243
6.113.2 Constructor & Destructor Documentation	243
6.113.2.1 SphereOverlapQueryParams()	243
6.113.3 Member Data Documentation	244
6.113.3.1 Center	244

6.113.3.2 QueryParams	244
6.113.3.3 Radius	244
6.113.3.4 StaticHitsCapacity	244
6.114 LagCompensationSettings Class Reference	244
6.114.1 Detailed Description	245
6.114.2 Member Data Documentation	245
6.114.2.1 CachedStaticCollidersSize	245
6.114.2.2 Enabled	245
6.114.2.3 HitboxBufferLengthInMs	245
6.114.2.4 HitboxDefaultCapacity	246
6.114.3 Property Documentation	246
6.114.3.1 ExpansionFactor	246
6.114.3.2 Optimize	246
6.115 LobbyInfo Class Reference	246
6.115.1 Detailed Description	246
6.115.2 Property Documentation	246
6.115.2.1 IsValid	247
6.115.2.2 Name	247
6.115.2.3 Region	247
6.116 LogSimpleUnity Class Reference	247
6.116.1 Detailed Description	247
6.117 NestedComponentUtilities Class Reference	247
6.117.1 Detailed Description	248
6.117.2 Member Function Documentation	249
6.117.2.1 EnsureRootComponentExists< T, TStopOn >()	249
6.117.2.2 FindObjectsOfTypeInOrder< T >() [1/2]	249
6.117.2.3 FindObjectsOfTypeInOrder< T >() [2/2]	250
6.117.2.4 FindObjectsOfTypeInOrder< T, TCast >() [1/2]	251
6.117.2.5 FindObjectsOfTypeInOrder< T, TCast >() [2/2]	251
6.117.2.6 GetNestedComponentInChildren< T, TStopOn >()	252
6.117.2.7 GetNestedComponentInParent< T, TStopOn >()	253
6.117.2.8 GetNestedComponentInParents< T, TStopOn >()	253
6.117.2.9 GetNestedComponentsInChildren< T >()	254
6.117.2.10 GetNestedComponentsInChildren< T, TSearch, TStop >()	254
6.117.2.11 GetNestedComponentsInChildren< T, TStopOn >()	254
6.117.2.12 GetNestedComponentsInParents< T >()	255
6.117.2.13 GetNestedComponentsInParents< T, TStop >()	255
6.117.2.14 GetParentComponent< T >()	255
6.118 NetworkArray< T > Struct Template Reference	255
6.118.1 Detailed Description	257
6.118.2 Constructor & Destructor Documentation	257
6.118.2.1 NetworkArray()	257

6.118.3 Member Function Documentation	257
6.118.3.1 Clear()	258
6.118.3.2 CopyFrom() [1/2]	258
6.118.3.3 CopyFrom() [2/2]	258
6.118.3.4 CopyTo() [1/3]	259
6.118.3.5 CopyTo() [2/3]	259
6.118.3.6 CopyTo() [3/3]	259
6.118.3.7 Get()	260
6.118.3.8 GetEnumerator()	260
6.118.3.9 operator NetworkArrayReadOnly< T >()	260
6.118.3.10 Set()	260
6.118.3.11 ToArray()	261
6.118.3.12 ToStringString()	261
6.118.3.13 ToReadOnly()	261
6.118.3.14 ToString()	261
6.118.4 Property Documentation	261
6.118.4.1 Length	261
6.118.4.2 this[int index]	261
6.119 NetworkArray< T >.Enumerator Struct Reference	262
6.119.1 Detailed Description	262
6.119.2 Constructor & Destructor Documentation	262
6.119.2.1 Enumerator()	262
6.119.3 Member Function Documentation	263
6.119.3.1 Dispose()	263
6.119.3.2 MoveNext()	263
6.119.3.3 Reset()	263
6.119.4 Property Documentation	263
6.119.4.1 Current [1/2]	263
6.119.4.2 Current [2/2]	264
6.120 NetworkArrayExtensions Class Reference	264
6.120.1 Detailed Description	264
6.120.2 Member Function Documentation	264
6.120.2.1 GetRef< T >()	264
6.120.2.2 IndexOf< T >()	265
6.121 NetworkArrayReadOnly< T > Struct Template Reference	265
6.121.1 Detailed Description	265
6.121.2 Property Documentation	266
6.121.2.1 Length	266
6.121.2.2 this[int index]	266
6.122 NetworkAssemblyIgnoreAttribute Class Reference	266
6.122.1 Detailed Description	266
6.123 NetworkAssemblyWeavedAttribute Class Reference	266

6.123.1 Detailed Description	266
6.124 NetworkBehaviour Class Reference	266
6.124.1 Detailed Description	270
6.124.2 Member Function Documentation	270
6.124.2.1 CopyBackingFieldsToState()	270
6.124.2.2 CopyStateFrom()	270
6.124.2.3 CopyStateToBackingFields()	270
6.124.2.4 Despawned()	271
6.124.2.5 FixedUpdateNetwork()	271
6.124.2.6 GetArrayReader< T >() [1/2]	271
6.124.2.7 GetArrayReader< T >() [2/2]	272
6.124.2.8 GetBehaviourReader< T >() [1/2]	272
6.124.2.9 GetBehaviourReader< T >() [2/2]	273
6.124.2.10 GetBehaviourReader< TBehaviour, TProperty >()	273
6.124.2.11 GetChangeDetector()	274
6.124.2.12 GetDictionaryReader< K, V >() [1/2]	274
6.124.2.13 GetDictionaryReader< K, V >() [2/2]	275
6.124.2.14 GetInput< T >() [1/2]	275
6.124.2.15 GetInput< T >() [2/2]	276
6.124.2.16 GetLinkListReader< T >() [1/2]	276
6.124.2.17 GetLinkListReader< T >() [2/2]	276
6.124.2.18 GetLocalAuthorityMask()	277
6.124.2.19GetPropertyReader< T >() [1/2]	277
6.124.2.20GetPropertyReader< T >() [2/2]	278
6.124.2.21GetPropertyReader< TBehaviour, TProperty >()	278
6.124.2.22 MakeInitializer< K, V >()	279
6.124.2.23 MakeInitializer< T >()	279
6.124.2.24 MakePtr< T >() [1/2]	279
6.124.2.25 MakePtr< T >() [2/2]	280
6.124.2.26 MakeRef< T >() [1/2]	280
6.124.2.27 MakeRef< T >() [2/2]	281
6.124.2.28 NetworkDeserialize()	281
6.124.2.29 NetworkSerialize() [1/2]	282
6.124.2.30 NetworkSerialize() [2/2]	282
6.124.2.31 NetworkUnwrap()	283
6.124.2.32 NetworkWrap()	283
6.124.2.33 operator NetworkBehaviourId()	283
6.124.2.34 ReinterpretState< T >()	285
6.124.2.35 ReplicateTo() [1/2]	285
6.124.2.36 ReplicateTo() [2/2]	286
6.124.2.37 ReplicateToAll()	286
6.124.2.38 ResetState()	286

6.124.2.39 Spawnsed()	287
6.124.2.40 TryGetSnapshotsBuffers()	287
6.124.3 Member Data Documentation	287
6.124.3.1 offset	287
6.124.4 Property Documentation	287
6.124.4.1 ChangedTick	288
6.124.4.2 DynamicWordCount	288
6.124.4.3 HasInputAuthority	288
6.124.4.4 HasStateAuthority	288
6.124.4.5 Id	288
6.124.4.6 IsProxy	288
6.124.4.7 StateBuffer	289
6.124.4.8 StateBufferIsValid	289
6.125 NetworkBehaviour.ArrayReader< T > Struct Template Reference	289
6.125.1 Detailed Description	289
6.125.2 Member Function Documentation	289
6.125.2.1 Read()	289
6.126 NetworkBehaviour.BehaviourReader< T > Struct Template Reference	290
6.126.1 Detailed Description	290
6.126.2 Member Function Documentation	290
6.126.2.1 Read()	290
6.126.3 Member Data Documentation	291
6.126.3.1 T	291
6.127 NetworkBehaviour.ChangeDetector Class Reference	291
6.127.1 Detailed Description	292
6.127.2 Member Enumeration Documentation	292
6.127.2.1 Source	292
6.127.3 Member Function Documentation	292
6.127.3.1 DetectChanges() [1/2]	292
6.127.3.2 DetectChanges() [2/2]	293
6.127.3.3 Init()	293
6.128 NetworkBehaviour.ChangeDetector.Enumerable Struct Reference	294
6.128.1 Detailed Description	294
6.128.2 Member Function Documentation	294
6.128.2.1 Changed()	294
6.128.2.2 GetEnumerator()	294
6.129 NetworkBehaviour.ChangeDetector.Enumerator Struct Reference	295
6.129.1 Detailed Description	295
6.129.2 Member Function Documentation	295
6.129.2.1 MoveNext()	295
6.129.2.2 Reset()	295
6.129.3 Property Documentation	296

6.129.3.1 Current	296
6.130 NetworkBehaviour.DictionaryReader< K, V > Struct Template Reference	296
6.130.1 Detailed Description	296
6.130.2 Member Function Documentation	296
6.130.2.1 Read()	296
6.131 NetworkBehaviour.LinkListReader< T > Struct Template Reference	297
6.131.1 Detailed Description	297
6.131.2 Member Function Documentation	297
6.131.2.1 Read()	297
6.132 NetworkBehaviour.PropertyReader< T > Struct Template Reference	298
6.132.1 Detailed Description	298
6.132.2 Constructor & Destructor Documentation	298
6.132.2.1 PropertyReader()	298
6.132.3 Member Function Documentation	299
6.132.3.1 Read()	299
6.132.4 Member Data Documentation	299
6.132.4.1 T	299
6.133 NetworkBehaviourBuffer Struct Reference	299
6.133.1 Detailed Description	300
6.133.2 Member Function Documentation	300
6.133.2.1 operator bool()	300
6.133.2.2 Read() [1/5]	301
6.133.2.3 Read() [2/5]	301
6.133.2.4 Read() [3/5]	301
6.133.2.5 Read() [4/5]	303
6.133.2.6 Read() [5/5]	303
6.133.2.7 Read< T >() [1/2]	303
6.133.2.8 Read< T >() [2/2]	305
6.133.2.9 ReinterpretState< T >()	305
6.133.3 Property Documentation	306
6.133.3.1 Length	306
6.133.3.2 this[int index]	306
6.133.3.3 Tick	306
6.133.3.4 Valid	307
6.134 NetworkBehaviourBufferInterpolator Struct Reference	307
6.134.1 Detailed Description	308
6.134.2 Constructor & Destructor Documentation	308
6.134.2.1 NetworkBehaviourBufferInterpolator()	308
6.134.3 Member Function Documentation	309
6.134.3.1 Angle() [1/2]	309
6.134.3.2 Angle() [2/2]	309
6.134.3.3 Bool() [1/2]	309

6.134.3.4 Bool() [2/2]	310
6.134.3.5 Float() [1/2]	310
6.134.3.6 Float() [2/2]	310
6.134.3.7 Int() [1/2]	312
6.134.3.8 Int() [2/2]	312
6.134.3.9 operator bool()	312
6.134.3.10 Quaternion() [1/2]	313
6.134.3.11 Quaternion() [2/2]	313
6.134.3.12 Select< T >() [1/2]	314
6.134.3.13 Select< T >() [2/2]	314
6.134.3.14 Vector2() [1/2]	315
6.134.3.15 Vector2() [2/2]	315
6.134.3.16 Vector3() [1/2]	315
6.134.3.17 Vector3() [2/2]	316
6.134.3.18 Vector4() [1/2]	316
6.134.3.19 Vector4() [2/2]	316
6.134.4 Member Data Documentation	317
6.134.4.1 Alpha	317
6.134.4.2 Behaviour	317
6.134.4.3 From	317
6.134.4.4 To	317
6.134.4.5 Valid	317
6.135 NetworkBehaviourId Struct Reference	318
6.135.1 Detailed Description	319
6.135.2 Member Function Documentation	319
6.135.2.1 Equals() [1/2]	319
6.135.2.2 Equals() [2/2]	319
6.135.2.3 GetHashCode()	319
6.135.2.4 operator"!=()"	320
6.135.2.5 operator==()	320
6.135.2.6 ToString()	321
6.135.3 Member Data Documentation	321
6.135.3.1 Behaviour	321
6.135.3.2 Object	321
6.135.3.3 SIZE	321
6.135.4 Property Documentation	321
6.135.4.1 IsValid	321
6.135.4.2 None	322
6.136 NetworkBehaviourUtils Class Reference	322
6.136.1 Detailed Description	323
6.136.2 Member Function Documentation	324
6.136.2.1 CloneArray< T >()	324

6.136.2.2 CopyFromNetworkArray< T >()	324
6.136.2.3 CopyFromNetworkDictionary< D, K, V >()	325
6.136.2.4 CopyFromNetworkList< T >()	325
6.136.2.5 GetMetaData()	326
6.136.2.6 GetRpcStaticIndexOrThrow()	326
6.136.2.7 GetStaticWordCount()	327
6.136.2.8 GetWordCount()	327
6.136.2.9 HasStaticWordCount()	328
6.136.2.10 InitializeNetworkArray< T >()	328
6.136.2.11 InitializeNetworkDictionary< D, K, V >()	329
6.136.2.12 InitializeNetworkList< T >()	329
6.136.2.13 InternalOnDestroy()	330
6.136.2.14 InternalOnDisable()	330
6.136.2.15 InternalOnEnable()	330
6.136.2.16 MakeSerializableDictionary< K, V >()	331
6.136.2.17 NotifyLocalSimulationNotAllowedToSendRpc()	331
6.136.2.18 NotifyLocalTargetedRpcCulled()	332
6.136.2.19 NotifyNetworkUnwrapFailed< T >()	332
6.136.2.20 NotifyNetworkWrapFailed< T >() [1/2]	332
6.136.2.21 NotifyNetworkWrapFailed< T >() [2/2]	333
6.136.2.22 NotifyRpcPayloadSizeExceeded()	333
6.136.2.23 NotifyRpcTargetUnreachable()	333
6.136.2.24 RegisterMetaData()	334
6.136.2.25 RegisterRpcInvokeDelegates()	334
6.136.2.26 ShouldRegisterRpcInvokeDelegates()	334
6.136.2.27 ThrowIfBehaviourNotInitialized()	335
6.136.2.28 TryGetRpcInvokeDelegateArray()	335
6.136.2.29 TryGetRpcStaticInvokeDelegate()	336
6.136.3 Member Data Documentation	336
6.136.3.1 InvokeRpc	336
6.137 NetworkBehaviourUtils.ArrayInitializer< T > Struct Template Reference	336
6.137.1 Detailed Description	336
6.137.2 Member Function Documentation	337
6.137.2.1 operator NetworkArray< T >()	337
6.137.2.2 operator NetworkLinkedList< T >()	337
6.138 NetworkBehaviourUtils.DictionaryInitializer< K, V > Struct Template Reference	338
6.138.1 Detailed Description	338
6.138.2 Member Function Documentation	338
6.138.2.1 operator NetworkDictionary< K, V >()	338
6.139 NetworkBehaviourUtils.MetaData Struct Reference	339
6.139.1 Detailed Description	339
6.140 NetworkBehaviourWeavedAttribute Class Reference	339

6.140.1 Detailed Description	339
6.140.2 Constructor & Destructor Documentation	339
6.140.2.1 NetworkBehaviourWeavedAttribute()	339
6.140.3 Property Documentation	340
6.140.3.1 WordCount	340
6.141 NetworkBool Struct Reference	340
6.141.1 Detailed Description	341
6.141.2 Constructor & Destructor Documentation	341
6.141.2.1 NetworkBool()	341
6.141.3 Member Function Documentation	341
6.141.3.1 Equals() [1/2]	341
6.141.3.2 Equals() [2/2]	341
6.141.3.3 GetHashCode()	342
6.141.3.4 operator bool()	342
6.141.3.5 operator NetworkBool()	342
6.141.3.6 ToString()	343
6.142 NetworkButtons Struct Reference	343
6.142.1 Detailed Description	344
6.142.2 Constructor & Destructor Documentation	344
6.142.2.1 NetworkButtons()	344
6.142.3 Member Function Documentation	344
6.142.3.1 Equals() [1/2]	345
6.142.3.2 Equals() [2/2]	345
6.142.3.3 GetHashCode()	345
6.142.3.4 GetPressed()	345
6.142.3.5 GetReleased()	346
6.142.3.6 IsSet()	346
6.142.3.7 IsSet< T >()	346
6.142.3.8 Set()	348
6.142.3.9 Set< T >()	348
6.142.3.10 SetAllDown()	349
6.142.3.11 SetAllUp()	349
6.142.3.12 SetDown()	349
6.142.3.13 SetDown< T >()	349
6.142.3.14 SetUp()	350
6.142.3.15 SetUp< T >()	350
6.142.3.16 WasPressed()	351
6.142.3.17 WasPressed< T >()	351
6.142.3.18 WasReleased()	352
6.142.3.19 WasReleased< T >()	352
6.142.4 Member Data Documentation	352
6.142.4.1 NetworkButtons	353

6.142.5 Property Documentation	353
6.142.5.1 Bits	353
6.143 NetworkConfiguration Class Reference	353
6.143.1 Detailed Description	354
6.143.2 Member Enumeration Documentation	354
6.143.2.1 ReliableDataTransfers	354
6.143.3 Member Function Documentation	354
6.143.3.1 Init()	354
6.143.4 Member Data Documentation	355
6.143.4.1 ConnectionShutdownTime	355
6.143.4.2 ConnectionTimeout	355
6.143.4.3 ReliableDataTransferModes	355
6.143.5 Property Documentation	355
6.143.5.1 ConnectAttempts	355
6.143.5.2 ConnectInterval	356
6.143.5.3 ConnectionDefaultRtt	356
6.143.5.4 ConnectionPingInterval	356
6.143.5.5 MtuDefault	356
6.143.5.6 SocketRecvBufferSize	356
6.143.5.7 SocketSendBufferSize	356
6.144 NetworkDelegates Class Reference	357
6.144.1 Detailed Description	357
6.145 NetworkDeserializeMethodAttribute Class Reference	357
6.145.1 Detailed Description	357
6.146 NetworkDictionary< K, V > Struct Template Reference	358
6.146.1 Detailed Description	359
6.146.2 Constructor & Destructor Documentation	360
6.146.2.1 NetworkDictionary()	360
6.146.3 Member Function Documentation	360
6.146.3.1 Add() [1/2]	360
6.146.3.2 Add() [2/2]	360
6.146.3.3 Clear()	361
6.146.3.4 ContainsKey()	361
6.146.3.5 ContainsValue()	361
6.146.3.6 Get()	361
6.146.3.7 GetEnumerator()	362
6.146.3.8 operator NetworkDictionaryReadOnly< K, V >()	362
6.146.3.9 Remove() [1/2]	362
6.146.3.10 Remove() [2/2]	362
6.146.3.11 Set()	363
6.146.3.12 ToReadOnly()	363
6.146.3.13 TryGet()	363

6.146.4 Member Data Documentation	364
6.146.4.1 META_WORD_COUNT	364
6.146.5 Property Documentation	364
6.146.5.1 Capacity	364
6.146.5.2 Count	364
6.146.5.3 this[K key]	364
6.147 NetworkDictionary< K, V >.Enumerator Struct Reference	364
6.147.1 Detailed Description	365
6.147.2 Member Function Documentation	365
6.147.2.1 Dispose()	365
6.147.2.2 MoveNext()	365
6.147.2.3 Reset()	366
6.147.3 Property Documentation	366
6.147.3.1 Current	366
6.148 NetworkDictionaryReadOnly< K, V > Struct Template Reference	366
6.148.1 Detailed Description	367
6.148.2 Member Function Documentation	367
6.148.2.1 Get()	367
6.148.2.2 TryGet()	367
6.148.3 Property Documentation	368
6.148.3.1 Capacity	368
6.148.3.2 Count	368
6.149 NetworkedAttribute Class Reference	368
6.149.1 Detailed Description	369
6.149.2 Constructor & Destructor Documentation	369
6.149.2.1 NetworkedAttribute()	369
6.149.3 Property Documentation	369
6.149.3.1 Default	369
6.150 NetworkedWeavedAttribute Class Reference	369
6.150.1 Detailed Description	370
6.150.2 Constructor & Destructor Documentation	370
6.150.2.1 NetworkedWeavedAttribute()	370
6.150.3 Property Documentation	370
6.150.3.1 WordCount	370
6.150.3.2 WordOffset	370
6.151 NetworkEvents Class Reference	370
6.151.1 Detailed Description	372
6.152 NetworkEvents.ConnectFailedEvent Class Reference	372
6.152.1 Detailed Description	372
6.153 NetworkEvents.ConnectRequestEvent Class Reference	372
6.153.1 Detailed Description	372
6.154 NetworkEvents.CustomAuthenticationResponse Class Reference	372

6.154.1 Detailed Description	372
6.155 NetworkEvents.DisconnectFromServerEvent Class Reference	372
6.155.1 Detailed Description	373
6.156 NetworkEvents.HostMigrationEvent Class Reference	373
6.156.1 Detailed Description	373
6.157 NetworkEvents.InputEvent Class Reference	373
6.157.1 Detailed Description	373
6.158 NetworkEvents.InputPlayerEvent Class Reference	373
6.158.1 Detailed Description	373
6.159 NetworkEvents.ObjectEvent Class Reference	373
6.159.1 Detailed Description	374
6.160 NetworkEvents.ObjectPlayerEvent Class Reference	374
6.160.1 Detailed Description	374
6.161 NetworkEvents.PlayerEvent Class Reference	374
6.161.1 Detailed Description	374
6.162 NetworkEvents.ReliableDataEvent Class Reference	374
6.162.1 Detailed Description	374
6.163 NetworkEvents.ReliableProgressEvent Class Reference	374
6.163.1 Detailed Description	375
6.164 NetworkEvents.RunnerEvent Class Reference	375
6.164.1 Detailed Description	375
6.165 NetworkEvents.SessionListUpdateEvent Class Reference	375
6.165.1 Detailed Description	375
6.166 NetworkEvents.ShutdownEvent Class Reference	375
6.166.1 Detailed Description	375
6.167 NetworkEvents.SimulationMessageEvent Class Reference	375
6.167.1 Detailed Description	376
6.168 NetworkId Struct Reference	376
6.168.1 Detailed Description	377
6.168.2 Member Function Documentation	377
6.168.2.1 CompareTo()	377
6.168.2.2 Equals() [1/2]	378
6.168.2.3 Equals() [2/2]	378
6.168.2.4 GetHashCode()	378
6.168.2.5 operator bool()	379
6.168.2.6 operator"!=()"	379
6.168.2.7 operator==()	379
6.168.2.8 Read()	379
6.168.2.9 ToNamePrefixString()	379
6.168.2.10 ToString()	380
6.168.2.11 Write() [1/2]	380
6.168.2.12 Write() [2/2]	380

6.168.3 Member Data Documentation	380
6.168.3.1 ALIGNMENT	380
6.168.3.2 BLOCK_SIZE	381
6.168.3.3 Raw	381
6.168.3.4 SIZE	381
6.168.4 Property Documentation	381
6.168.4.1 Comparer	381
6.168.4.2 IsReserved	381
6.168.4.3 IsValid	381
6.169 NetworkId.EqualityComparer Class Reference	382
6.169.1 Detailed Description	382
6.169.2 Member Function Documentation	382
6.169.2.1 Equals()	382
6.169.2.2 GetHashCode()	382
6.170 NetworkInput Struct Reference	382
6.170.1 Detailed Description	383
6.170.2 Member Function Documentation	383
6.170.2.1 Convert< T >()	383
6.170.2.2 Get< T >()	384
6.170.2.3 Is< T >()	384
6.170.2.4 Set< T >()	384
6.170.2.5 TryGet< T >()	384
6.170.2.6 TrySet< T >()	385
6.170.3 Property Documentation	385
6.170.3.1 Data	385
6.170.3.2 IsValid	385
6.170.3.3 Type	385
6.170.3.4 WordCount	385
6.171 NetworkInputUtils Class Reference	386
6.171.1 Detailed Description	386
6.171.2 Member Function Documentation	386
6.171.2.1 GetMaxWordCount()	386
6.171.2.2 GetType()	386
6.171.2.3 GetTypeKey()	387
6.171.2.4 GetWordCount()	387
6.172 NetworkInputWeavedAttribute Class Reference	387
6.172.1 Detailed Description	388
6.172.2 Constructor & Destructor Documentation	388
6.172.2.1 NetworkInputWeavedAttribute()	388
6.172.3 Property Documentation	388
6.172.3.1 WordCount	388
6.173 NetworkLinkedList< T > Struct Template Reference	388

6.173.1 Detailed Description	390
6.173.2 Constructor & Destructor Documentation	390
6.173.2.1 NetworkLinkedList()	391
6.173.3 Member Function Documentation	391
6.173.3.1 Add() [1/2]	391
6.173.3.2 Add() [2/2]	391
6.173.3.3 Clear()	392
6.173.3.4 Contains() [1/2]	392
6.173.3.5 Contains() [2/2]	392
6.173.3.6 Get()	392
6.173.3.7 GetEnumerator()	392
6.173.3.8 IndexOf() [1/2]	392
6.173.3.9 IndexOf() [2/2]	392
6.173.3.10 Remap()	393
6.173.3.11 Remove() [1/2]	393
6.173.3.12 Remove() [2/2]	393
6.173.3.13 Set()	394
6.173.4 Member Data Documentation	394
6.173.4.1 ELEMENT_WORDS	394
6.173.4.2 META_WORDS	394
6.173.5 Property Documentation	394
6.173.5.1 Capacity	394
6.173.5.2 Count	394
6.173.5.3 this[int index]	395
6.174 NetworkLinkedList< T >.Enumerator Struct Reference	395
6.174.1 Detailed Description	395
6.174.2 Member Function Documentation	395
6.174.2.1 Dispose()	396
6.174.2.2 MoveNext()	396
6.174.2.3 Reset()	396
6.174.3 Property Documentation	396
6.174.3.1 Current	396
6.175 NetworkLinkedListReadOnly< T > Struct Template Reference	397
6.175.1 Detailed Description	398
6.175.2 Member Function Documentation	399
6.175.2.1 Contains() [1/2]	399
6.175.2.2 Contains() [2/2]	399
6.175.2.3 Get()	399
6.175.2.4 IndexOf() [1/2]	399
6.175.2.5 IndexOf() [2/2]	399
6.175.3 Member Data Documentation	400
6.175.3.1 ELEMENT_WORDS	400

6.175.3.2 META_WORDS	400
6.175.4 Property Documentation	400
6.175.4.1 Capacity	400
6.175.4.2 Count	400
6.175.4.3 this[int index]	401
6.176 NetworkLoadSceneParameters Struct Reference	401
6.176.1 Detailed Description	402
6.176.2 Member Function Documentation	402
6.176.2.1 Equals() [1/2]	402
6.176.2.2 Equals() [2/2]	402
6.176.2.3 GetHashCode()	402
6.176.2.4 operator"!="()	403
6.176.2.5 operator==()	403
6.176.2.6 ToString()	403
6.176.3 Member Data Documentation	403
6.176.3.1 LoadId	404
6.176.4 Property Documentation	404
6.176.4.1 IsActiveOnLoad	404
6.176.4.2 IsLocalPhysics2D	404
6.176.4.3 IsLocalPhysics3D	404
6.176.4.4 IsSingleLoad	404
6.176.4.5 LoadSceneMode	404
6.176.4.6 LoadSceneParameters	405
6.176.4.7 LocalPhysicsMode	405
6.177 NetworkMecanimAnimator Class Reference	405
6.177.1 Detailed Description	406
6.177.2 Member Function Documentation	406
6.177.2.1 SetTrigger() [1/2]	406
6.177.2.2 SetTrigger() [2/2]	406
6.177.3 Member Data Documentation	407
6.177.3.1 Animator	407
6.177.3.2 ApplyTiming	407
6.177.4 Property Documentation	407
6.177.4.1 DynamicWordCount	407
6.178 NetworkObject Class Reference	408
6.178.1 Detailed Description	410
6.178.2 Member Function Documentation	410
6.178.2.1 AssignInputAuthority()	410
6.178.2.2 Awake()	410
6.178.2.3 CopyStateFrom() [1/2]	410
6.178.2.4 CopyStateFrom() [2/2]	411
6.178.2.5 GetLocalAuthorityMask()	411

6.178.2.6 GetWordCount()	411
6.178.2.7 NetworkUnwrap()	412
6.178.2.8 NetworkWrap() [1/2]	412
6.178.2.9 NetworkWrap() [2/2]	412
6.178.2.10 OnDestroy()	413
6.178.2.11 operator NetworkId()	413
6.178.2.12 PriorityLevelDelegate()	413
6.178.2.13 ReleaseStateAuthority()	414
6.178.2.14 RemoveInputAuthority()	414
6.178.2.15 ReplicateToDelegate()	414
6.178.2.16 RequestStateAuthority()	414
6.178.2.17 SetPlayerAlwaysInterested()	414
6.178.3 Member Data Documentation	415
6.178.3.1 Flags	415
6.178.3.2 IsResume	415
6.178.3.3 NestedObjects	415
6.178.3.4 NetworkedBehaviours	415
6.178.3.5 NetworkTypeId	415
6.178.3.6 PriorityCallback	416
6.178.3.7 ReplicateTo	416
6.178.3.8 SortKey	416
6.178.4 Property Documentation	416
6.178.4.1 HasInputAuthority	416
6.178.4.2 HasStateAuthority	416
6.178.4.3 Id	416
6.178.4.4 InputAuthority	417
6.178.4.5 IsInSimulation	417
6.178.4.6 IsProxy	417
6.178.4.7 IsSpawnable	417
6.178.4.8 IsValid	417
6.178.4.9 LastReceiveTick	417
6.178.4.10 Name	418
6.178.4.11 RenderSource	418
6.178.4.12 RenderTime	418
6.178.4.13 RenderTimeframe	418
6.178.4.14 Runner	418
6.178.4.15 StateAuthority	418
6.179 NetworkObjectFlagsExtensions Class Reference	419
6.179.1 Detailed Description	419
6.179.2 Member Function Documentation	419
6.179.2.1 GetVersion()	419
6.179.2.2 IsIgnored()	419

6.179.2.3 IsVersionCurrent()	421
6.179.2.4 SetCurrentVersion()	421
6.179.2.5 SetIgnored()	421
6.180 NetworkObjectGuid Struct Reference	422
6.180.1 Detailed Description	423
6.180.2 Constructor & Destructor Documentation	423
6.180.2.1 NetworkObjectGuid() [1/4]	423
6.180.2.2 NetworkObjectGuid() [2/4]	424
6.180.2.3 NetworkObjectGuid() [3/4]	424
6.180.2.4 NetworkObjectGuid() [4/4]	424
6.180.3 Member Function Documentation	425
6.180.3.1 CompareTo()	425
6.180.3.2 Equals() [1/2]	425
6.180.3.3 Equals() [2/2]	425
6.180.3.4 GetHashCode()	426
6.180.3.5 operator Guid()	426
6.180.3.6 operator NetworkObjectGuid()	426
6.180.3.7 operator NetworkPrefabRef()	427
6.180.3.8 operator"!=="	427
6.180.3.9 operator==()	427
6.180.3.10 Parse()	427
6.180.3.11 ToString() [1/2]	428
6.180.3.12 ToString() [2/2]	428
6.180.3.13 ToUnityGuidString()	428
6.180.3.14 TryParse()	428
6.180.4 Member Data Documentation	429
6.180.4.1 ALIGNMENT	429
6.180.4.2 RawGuidValue	429
6.180.4.3 SIZE	429
6.180.5 Property Documentation	429
6.180.5.1 Empty	429
6.180.5.2 IsValid	429
6.181 NetworkObjectGuid.EqualityComparer Class Reference	430
6.181.1 Detailed Description	430
6.181.2 Member Function Documentation	430
6.181.2.1 Equals()	430
6.181.2.2 GetHashCode()	430
6.182 NetworkObjectHeader Struct Reference	430
6.182.1 Detailed Description	432
6.182.2 Member Function Documentation	432
6.182.2.1 Equals() [1/2]	432
6.182.2.2 Equals() [2/2]	432

6.182.2.3 GetBehaviourChangedTickArray()	432
6.182.2.4 GetDataPointer()	433
6.182.2.5 GetDataWordCount()	433
6.182.2.6 GetHashCode()	433
6.182.2.7 GetMainNetworkTRSPData()	434
6.182.2.8 HasMainNetworkTRSP()	434
6.182.2.9 operator"!="()	434
6.182.2.10 operator=="()	435
6.182.2.11 ToString()	435
6.182.3 Member Data Documentation	435
6.182.3.1 _reserved	435
6.182.3.2 BehaviourCount	435
6.182.3.3 Flags	435
6.182.3.4 Id	436
6.182.3.5 InputAuthority	436
6.182.3.6 NestingKey	436
6.182.3.7 NestingRoot	436
6.182.3.8 PLAYER_DATA_WORD	436
6.182.3.9 SIZE	436
6.182.3.10 StateAuthority	437
6.182.3.11 Type	437
6.182.3.12 WordCount	437
6.182.3.13 WORDS	437
6.182.4 Property Documentation	437
6.182.4.1 ByteCount	437
6.183 NetworkObjectHeaderPtr Struct Reference	437
6.183.1 Detailed Description	438
6.183.2 Member Data Documentation	438
6.183.2.1 Ptr	438
6.183.3 Property Documentation	438
6.183.3.1 Id	438
6.183.3.2 Type	438
6.184 NetworkObjectInitializerUnity Class Reference	439
6.184.1 Detailed Description	439
6.184.2 Member Function Documentation	439
6.184.2.1 InitializeNetworkState()	439
6.185 NetworkObjectMeta Class Reference	439
6.185.1 Detailed Description	440
6.185.2 Property Documentation	440
6.185.2.1 Id	440
6.185.2.2 InputAuthority	440
6.185.2.3 StateAuthority	440

6.185.2.4 Type	440
6.186 NetworkObjectNestingKey Struct Reference	440
6.186.1 Detailed Description	441
6.186.2 Constructor & Destructor Documentation	441
6.186.2.1 NetworkObjectNestingKey()	441
6.186.3 Member Function Documentation	442
6.186.3.1 Equals() [1/2]	442
6.186.3.2 Equals() [2/2]	442
6.186.3.3 GetHashCode()	442
6.186.3.4 ToString()	443
6.186.4 Member Data Documentation	443
6.186.4.1 ALIGNMENT	443
6.186.4.2 SIZE	443
6.186.4.3 Value	443
6.186.5 Property Documentation	443
6.186.5.1 IsNone	444
6.186.5.2 IsValid	444
6.187 NetworkObjectNestingKey.EqualityComparer Class Reference	444
6.187.1 Detailed Description	444
6.187.2 Member Function Documentation	444
6.187.2.1 Equals()	445
6.187.2.2 GetHashCode()	445
6.188 NetworkObjectPrefabData Class Reference	445
6.188.1 Detailed Description	445
6.188.2 Member Data Documentation	445
6.188.2.1 Guid	445
6.189 NetworkObjectProviderDummy Class Reference	446
6.189.1 Detailed Description	446
6.190 NetworkObjectReleaseContext Struct Reference	446
6.190.1 Detailed Description	446
6.190.2 Constructor & Destructor Documentation	447
6.190.2.1 NetworkObjectReleaseContext()	447
6.190.3 Member Function Documentation	447
6.190.3.1 ToString()	447
6.190.4 Member Data Documentation	447
6.190.4.1 IsBeingDestroyed	447
6.190.4.2 IsNestedObject	448
6.190.4.3 Object	448
6.190.4.4 TypId	448
6.191 NetworkObjectSortKeyComparer Class Reference	448
6.191.1 Detailed Description	448
6.191.2 Member Function Documentation	448

6.191.2.1 Compare()	448
6.191.3 Member Data Documentation	449
6.191.3.1 Instance	449
6.192 NetworkObjectSpawnException Class Reference	449
6.192.1 Detailed Description	449
6.192.2 Constructor & Destructor Documentation	449
6.192.2.1 NetworkObjectSpawnException()	449
6.192.3 Property Documentation	450
6.192.3.1 Message	450
6.192.3.2 Status	450
6.192.3.3 Typeld	450
6.193 NetworkObjectTypeld Struct Reference	450
6.193.1 Detailed Description	452
6.193.2 Member Function Documentation	452
6.193.2.1 Equals() [1/2]	452
6.193.2.2 Equals() [2/2]	453
6.193.2.3 FromCustom()	453
6.193.2.4 FromPrefabId()	453
6.193.2.5 FromSceneRefAndObjectIndex()	454
6.193.2.6 FromStruct()	454
6.193.2.7 GetHashCode()	455
6.193.2.8 operator NetworkObjectTypeld()	455
6.193.2.9 operator"!="()	455
6.193.2.10 operator==()	455
6.193.2.11 ToString()	456
6.193.3 Member Data Documentation	456
6.193.3.1 _value0	456
6.193.3.2 _value1	456
6.193.3.3 ALIGNMENT	456
6.193.3.4 MAX_SCENE_OBJECT_INDEX	456
6.193.3.5 SIZE	456
6.193.4 Property Documentation	456
6.193.4.1 AsCustom	456
6.193.4.2 AsInternalStructId	457
6.193.4.3 AsPrefabId	457
6.193.4.4 AsSceneObjectId	457
6.193.4.5 Comparer	458
6.193.4.6 IsCustom	458
6.193.4.7 IsNone	458
6.193.4.8 IsPrefab	458
6.193.4.9 IsSceneObject	458
6.193.4.10 IsStruct	458

6.193.4.11 IsValid	459
6.193.4.12 Kind	459
6.193.4.13 PlayerData	459
6.194 NetworkObjectTypeld.EqualityComparer Class Reference	459
6.194.1 Detailed Description	459
6.194.2 Member Function Documentation	459
6.194.2.1 Equals()	460
6.194.2.2 GetHashCode()	460
6.195 NetworkPhysicsInfo Struct Reference	461
6.195.1 Detailed Description	461
6.195.2 Member Data Documentation	461
6.195.2.1 SIZE	461
6.195.2.2 TimeScale	461
6.195.2.3 WORD_COUNT	462
6.196 NetworkPrefabAcquireContext Struct Reference	462
6.196.1 Detailed Description	462
6.196.2 Constructor & Destructor Documentation	462
6.196.2.1 NetworkPrefabAcquireContext()	462
6.196.3 Member Data Documentation	463
6.196.3.1 DontDestroyOnLoad	463
6.196.3.2 IsSynchronous	463
6.196.3.3 Meta	463
6.196.3.4 PrefabId	463
6.196.4 Property Documentation	463
6.196.4.1 Data	463
6.196.4.2 HasHeader	464
6.197 NetworkPrefabAttribute Class Reference	464
6.197.1 Detailed Description	464
6.198 NetworkPrefabId Struct Reference	464
6.198.1 Detailed Description	466
6.198.2 Member Function Documentation	466
6.198.2.1 CompareTo() [1/2]	466
6.198.2.2 CompareTo() [2/2]	466
6.198.2.3 Equals() [1/2]	466
6.198.2.4 Equals() [2/2]	466
6.198.2.5 FromIndex()	467
6.198.2.6 FromRaw()	467
6.198.2.7 GetHashCode()	467
6.198.2.8 operator"!=()"	467
6.198.2.9 operator==()	467
6.198.2.10 ToString() [1/2]	468
6.198.2.11 ToString() [2/2]	468

6.198.3 Member Data Documentation	468
6.198.3.1 ALIGNMENT	468
6.198.3.2 MAX_INDEX	468
6.198.3.3 RawValue	468
6.198.3.4 SIZE	468
6.198.4 Property Documentation	469
6.198.4.1 AsIndex	469
6.198.4.2 IsNone	469
6.198.4.3 IsValid	469
6.199 NetworkPrefabId.EqualityComparer Class Reference	469
6.199.1 Detailed Description	469
6.199.2 Member Function Documentation	470
6.199.2.1 Equals()	470
6.199.2.2 GetHashCode()	470
6.200 NetworkPrefabInfo Struct Reference	470
6.200.1 Detailed Description	470
6.200.2 Member Data Documentation	471
6.200.2.1 Header	471
6.200.2.2 IsSynchronous	471
6.200.2.3 Prefab	471
6.200.3 Property Documentation	471
6.200.3.1 Data	471
6.200.3.2 HasHeader	471
6.201 NetworkPrefabRef Struct Reference	472
6.201.1 Detailed Description	473
6.201.2 Constructor & Destructor Documentation	473
6.201.2.1 NetworkPrefabRef() [1/4]	473
6.201.2.2 NetworkPrefabRef() [2/4]	473
6.201.2.3 NetworkPrefabRef() [3/4]	474
6.201.2.4 NetworkPrefabRef() [4/4]	474
6.201.3 Member Function Documentation	474
6.201.3.1 CompareTo()	474
6.201.3.2 Equals() [1/2]	475
6.201.3.3 Equals() [2/2]	475
6.201.3.4 GetHashCode()	475
6.201.3.5 operator Guid()	476
6.201.3.6 operator NetworkObjectGuid()	476
6.201.3.7 operator NetworkPrefabRef()	476
6.201.3.8 operator"!=()"	477
6.201.3.9 operator==()	477
6.201.3.10 Parse()	477
6.201.3.11 ToString() [1/2]	478

6.201.3.12 <code>ToString()</code> [2/2]	478
6.201.3.13 <code>ToUnityGuidString()</code>	478
6.201.3.14 <code>TryParse()</code>	478
6.201.4 Member Data Documentation	478
6.201.4.1 ALIGNMENT	479
6.201.4.2 RawGuidValue	479
6.201.4.3 SIZE	479
6.201.5 Property Documentation	479
6.201.5.1 Empty	479
6.201.5.2 IsValid	479
6.202 NetworkPrefabRef.EqualityComparer Class Reference	479
6.202.1 Detailed Description	480
6.202.2 Member Function Documentation	480
6.202.2.1 Equals()	480
6.202.2.2 GetHashCode()	480
6.203 NetworkPrefabTable Class Reference	480
6.203.1 Detailed Description	482
6.203.2 Member Function Documentation	482
6.203.2.1 AddInstance()	482
6.203.2.2 AddSource()	482
6.203.2.3 Clear()	482
6.203.2.4 Contains()	483
6.203.2.5 GetEntries()	483
6.203.2.6 GetGuid()	483
6.203.2.7 GetId()	484
6.203.2.8 GetInstancesCount()	484
6.203.2.9 GetSource() [1/2]	484
6.203.2.10 GetSource() [2/2]	485
6.203.2.11 IsAcquired()	485
6.203.2.12 Load()	485
6.203.2.13 RemoveInstance()	486
6.203.2.14 TryAddSource()	486
6.203.2.15 Unload()	487
6.203.2.16 UnloadAll()	487
6.203.2.17 UnloadUnreferenced()	487
6.203.3 Member Data Documentation	488
6.203.3.1 Options	488
6.203.4 Property Documentation	488
6.203.4.1 Prefabs	488
6.203.4.2 Version	488
6.204 NetworkPrefabTableOptions Struct Reference	488
6.204.1 Detailed Description	489

6.204.2 Member Data Documentation	489
6.204.2.1 Default	489
6.204.2.2 UnloadPrefabOnReleasingLastInstance	489
6.204.2.3 UnloadUnusedPrefabsOnShutdown	489
6.205 NetworkProjectConfig Class Reference	489
6.205.1 Detailed Description	491
6.205.2 Member Enumeration Documentation	491
6.205.2.1 PeerModes	491
6.205.2.2 ReplicationFeatures	492
6.205.3 Member Function Documentation	492
6.205.3.1 Deserialize()	492
6.205.3.2 GetExecutionOrder()	493
6.205.3.3 Serialize()	493
6.205.3.4 ToString()	493
6.205.3.5 UnloadGlobal()	493
6.205.4 Member Data Documentation	494
6.205.4.1 AssembliesToWeave	494
6.205.4.2 BuildTypes	494
6.205.4.3 CheckNetworkedPropertiesBeingEmpty	494
6.205.4.4 CheckRpcAttributeUsage	494
6.205.4.5 CurrentTypeId	494
6.205.4.6 CurrentVersion	495
6.205.4.7 DefaultResourceName	495
6.205.4.8 EncryptionConfig	495
6.205.4.9 EnqueueIncompleteSynchronousSpawns	495
6.205.4.10 Heap	495
6.205.4.11 HideNetworkObjectInactivityGuard	495
6.205.4.12 HostMigration	496
6.205.4.13 InvokeRenderInBatchMode	496
6.205.4.14 LagCompensation	496
6.205.4.15 Network	496
6.205.4.16 NetworkConditions	496
6.205.4.17 NetworkIdIsObjectName	496
6.205.4.18 NullChecksForNetworkedProperties	497
6.205.4.19 PeerMode	497
6.205.4.20 PrefabTable	497
6.205.4.21 Simulation	497
6.205.4.22 TimeSynchronizationOverride	497
6.205.4.23 Typeld	497
6.205.4.24 UseSerializableDictionary	498
6.205.4.25 Version	498
6.205.5 Property Documentation	498

6.205.5.1 Global	498
6.206 NetworkProjectConfigAsset Class Reference	498
6.206.1 Detailed Description	499
6.206.2 Member Function Documentation	499
6.206.2.1 TryGetGlobal()	499
6.206.2.2 UnloadGlobal()	500
6.206.3 Member Data Documentation	500
6.206.3.1 BehaviourMeta	500
6.206.3.2 Config	500
6.206.3.3 PrefabOptions	500
6.206.3.4 Prefabs	500
6.206.4 Property Documentation	500
6.206.4.1 Global	501
6.206.4.2 IsGlobalLoaded	501
6.207 NetworkProjectConfigAsset.SerializeableSimulationBehaviourMeta Struct Reference	501
6.207.1 Detailed Description	501
6.207.2 Member Data Documentation	501
6.207.2.1 ExecutionOrder	501
6.207.2.2 Type	502
6.208 NetworkRNG Struct Reference	502
6.208.1 Detailed Description	503
6.208.2 Constructor & Destructor Documentation	503
6.208.2.1 NetworkRNG()	503
6.208.3 Member Function Documentation	503
6.208.3.1 Next()	503
6.208.3.2 NextExclusive()	504
6.208.3.3 NextInt32()	504
6.208.3.4 NextSingle()	504
6.208.3.5 NextSingleExclusive()	504
6.208.3.6 NextUInt32()	505
6.208.3.7 RangeExclusive() [1/2]	505
6.208.3.8 RangeExclusive() [2/2]	505
6.208.3.9 RangeInclusive() [1/4]	505
6.208.3.10 RangeInclusive() [2/4]	506
6.208.3.11 RangeInclusive() [3/4]	506
6.208.3.12 RangeInclusive() [4/4]	506
6.208.3.13 ToString()	506
6.208.4 Member Data Documentation	506
6.208.4.1 MAX	506
6.208.4.2 SIZE	507
6.208.5 Property Documentation	507
6.208.5.1 Peek	507

6.209 NetworkRpcStaticWeavedInvokerAttribute Class Reference	507
6.209.1 Detailed Description	507
6.209.2 Constructor & Destructor Documentation	507
6.209.2.1 NetworkRpcStaticWeavedInvokerAttribute()	507
6.209.3 Property Documentation	508
6.209.3.1 Key	508
6.210 NetworkRpcWeavedInvokerAttribute Class Reference	508
6.210.1 Detailed Description	508
6.210.2 Constructor & Destructor Documentation	508
6.210.2.1 NetworkRpcWeavedInvokerAttribute()	508
6.210.3 Property Documentation	509
6.210.3.1 Key	509
6.210.3.2 Sources	509
6.210.3.3 Targets	509
6.211 NetworkRunner Class Reference	509
6.211.1 Detailed Description	518
6.211.2 Member Enumeration Documentation	518
6.211.2.1 BuildTypes	518
6.211.2.2 States	518
6.211.3 Member Function Documentation	519
6.211.3.1 AddCallbacks()	519
6.211.3.2 AddGlobal()	519
6.211.3.3 AddPlayerAreaOfInterest()	519
6.211.3.4 Attach() [1/2]	519
6.211.3.5 Attach() [2/2]	520
6.211.3.6 ClearPlayerAreaOfInterest()	520
6.211.3.7 Despawn()	520
6.211.3.8 DestroySingleton< T >()	521
6.211.3.9 Disconnect()	521
6.211.3.10 EnsureRunnerScenesActive()	521
6.211.3.11 Exists() [1/2]	522
6.211.3.12 Exists() [2/2]	522
6.211.3.13 FindObject()	522
6.211.3.14 GetAllBehaviours()	523
6.211.3.15 GetAllBehaviours< T >() [1/2]	523
6.211.3.16 GetAllBehaviours< T >() [2/2]	523
6.211.3.17 GetAreaOfInterestGizmoData()	524
6.211.3.18 GetInputForPlayer< T >()	524
6.211.3.19 GetInstancesEnumerator()	524
6.211.3.20 GetInterfaceListHead()	524
6.211.3.21 GetInterfaceListNext()	525
6.211.3.22 GetInterfaceListPrev()	525

6.211.3.23 GetInterfaceListsCount()	526
6.211.3.24 GetPhysicsScene()	526
6.211.3.25 GetPhysicsScene2D()	526
6.211.3.26 GetPlayerActorId()	526
6.211.3.27 GetPlayerConnectionToken()	527
6.211.3.28 GetPlayerConnectionType()	527
6.211.3.29 GetPlayerObject()	527
6.211.3.30 GetPlayerRtt()	528
6.211.3.31 GetPlayerUserId()	528
6.211.3.32 GetRawInputForPlayer()	528
6.211.3.33 GetResumeSnapshotNetworkObjects()	529
6.211.3.34 GetResumeSnapshotNetworkSceneObjects()	529
6.211.3.35 GetRpcTargetStatus()	529
6.211.3.36 GetRunnerForObject()	529
6.211.3.37 GetRunnerForScene()	530
6.211.3.38 GetSingleton< T >()	530
6.211.3.39 HasSingleton< T >()	530
6.211.3.40 InstantiateInRunnerScene() [1/2]	531
6.211.3.41 InstantiateInRunnerScene() [2/2]	531
6.211.3.42 InstantiateInRunnerScene< T >() [1/2]	531
6.211.3.43 InstantiateInRunnerScene< T >() [2/2]	531
6.211.3.44 InvokeSceneLoadDone()	532
6.211.3.45 InvokeSceneLoadStart()	532
6.211.3.46 IsInterestedIn()	532
6.211.3.47 IsPlayerValid()	532
6.211.3.48 JoinSessionLobby()	533
6.211.3.49 LoadScene() [1/3]	533
6.211.3.50 LoadScene() [2/3]	534
6.211.3.51 LoadScene() [3/3]	534
6.211.3.52 MakeDontDestroyOnLoad()	535
6.211.3.53 MoveGameObjectToSameScene()	535
6.211.3.54 MoveGameObjectToScene()	535
6.211.3.55 MoveToRunnerScene()	536
6.211.3.56 MoveToRunnerScene< T >()	536
6.211.3.57 ObjectDelegate()	536
6.211.3.58 OnBeforeSpawned()	537
6.211.3.59 PushHostMigrationSnapshot()	537
6.211.3.60 RegisterSceneObjects()	537
6.211.3.61 RemoveCallbacks()	538
6.211.3.62 RemoveGlobal()	538
6.211.3.63 RenderInternal()	538
6.211.3.64 SendReliableDataToPlayer()	538

6.211.3.65 SendReliableDataToServer()	539
6.211.3.66 SendRpc() [1/2]	539
6.211.3.67 SendRpc() [2/2]	539
6.211.3.68 SetAreaOfInterestCellSize()	539
6.211.3.69 SetAreaOfInterestGrid()	541
6.211.3.70 SetBehaviourReplicateTo()	541
6.211.3.71 SetBehaviourReplicateToAll()	542
6.211.3.72 SetIsSimulated()	542
6.211.3.73 SetMasterClient()	542
6.211.3.74 SetPlayerAlwaysInterested()	543
6.211.3.75 SetPlayerObject()	543
6.211.3.76 Shutdown()	543
6.211.3.77 SinglePlayerContinue()	543
6.211.3.78 SinglePlayerPause() [1/2]	544
6.211.3.79 SinglePlayerPause() [2/2]	544
6.211.3.80 Spawn() [1/5]	544
6.211.3.81 Spawn() [2/5]	544
6.211.3.82 Spawn() [3/5]	545
6.211.3.83 Spawn() [4/5]	546
6.211.3.84 Spawn() [5/5]	546
6.211.3.85 Spawn< T >()	547
6.211.3.86 SpawnAsync() [1/5]	547
6.211.3.87 SpawnAsync() [2/5]	548
6.211.3.88 SpawnAsync() [3/5]	549
6.211.3.89 SpawnAsync() [4/5]	549
6.211.3.90 SpawnAsync() [5/5]	550
6.211.3.91 SpawnAsync< T >()	551
6.211.3.92 StartGame()	551
6.211.3.93 TryFindBehaviour()	552
6.211.3.94 TryFindBehaviour< T >()	552
6.211.3.95 TryFindObject()	553
6.211.3.96 TryGetFusionStatistics()	553
6.211.3.97 TryGetInputForPlayer< T >()	553
6.211.3.98 TryGetNetworkedBehaviourFromNetworkedObjectRef< T >()	554
6.211.3.99 TryGetNetworkedBehaviourId()	554
6.211.3.100 TryGetObjectRefFromNetworkedBehaviour()	555
6.211.3.101 TryGetPhysicsInfo()	555
6.211.3.102 TryGetPlayerObject()	555
6.211.3.103 TryGetSceneInfo()	556
6.211.3.104 TrySetPhysicsInfo()	556
6.211.3.105 TrySpawn() [1/5]	557
6.211.3.106 TrySpawn() [2/5]	557

6.211.3.107 TrySpawn() [3/5]	558
6.211.3.108 TrySpawn() [4/5]	558
6.211.3.109 TrySpawn() [5/5]	559
6.211.3.110 TrySpawn< T >()	560
6.211.3.111 UnloadScene()	560
6.211.3.112 UpdateInternal()	562
6.211.4 Property Documentation	562
6.211.4.1 ActivePlayers	562
6.211.4.2 AuthenticationValues	562
6.211.4.3 BuildType	562
6.211.4.4 CanSpawn	563
6.211.4.5 Config	563
6.211.4.6 CurrentConnectionType	563
6.211.4.7 DeltaTime	563
6.211.4.8 GameMode	563
6.211.4.9 Instances	563
6.211.4.10 IsClient	564
6.211.4.11 IsCloudReady	564
6.211.4.12 IsConnectedToServer	564
6.211.4.13 IsFirstTick	564
6.211.4.14 IsForward	564
6.211.4.15 IsLastTick	564
6.211.4.16 IsPlayer	565
6.211.4.17 IsResimulation	565
6.211.4.18 IsResume	565
6.211.4.19 IsRunning	565
6.211.4.20 IsSceneAuthority	565
6.211.4.21 IsSceneManagerBusy	565
6.211.4.22 IsServer	566
6.211.4.23 IsSharedModeMasterClient	566
6.211.4.24 IsShutdown	566
6.211.4.25 IsSinglePlayer	566
6.211.4.26 IsStarting	566
6.211.4.27 LagCompensation	566
6.211.4.28 LatestServerTick	567
6.211.4.29 LobbyInfo	567
6.211.4.30 LocalAlpha	567
6.211.4.31 LocalPlayer	567
6.211.4.32 LocalRenderTime	567
6.211.4.33 Mode	567
6.211.4.34 NATType	568
6.211.4.35 ObjectProvider	568

6.211.4.36 Prefabs	568
6.211.4.37 ProvideInput	568
6.211.4.38 RemoteRenderTime	568
6.211.4.39 SceneManager	568
6.211.4.40 SessionInfo	569
6.211.4.41 SimulationTime	569
6.211.4.42 SimulationUnityScene	569
6.211.4.43 Stage	569
6.211.4.44 State	569
6.211.4.45 Tick	569
6.211.4.46 TickRate	570
6.211.4.47 TicksExecuted	570
6.211.4.48 Topology	570
6.211.4.49 UserId	570
6.211.5 Event Documentation	570
6.211.5.1 ObjectAcquired	570
6.212 NetworkRunnerCallbackArgs Class Reference	570
6.212.1 Detailed Description	571
6.213 NetworkRunnerCallbackArgs.ConnectRequest Class Reference	571
6.213.1 Detailed Description	571
6.213.2 Member Function Documentation	571
6.213.2.1 Accept()	571
6.213.2.2 Refuse()	572
6.213.2.3 Waiting()	572
6.213.3 Property Documentation	572
6.213.3.1 RemoteAddress	572
6.214 NetworkRunnerUpdaterDefault Class Reference	572
6.214.1 Detailed Description	573
6.214.2 Member Function Documentation	573
6.214.2.1 RegisterInPlayerLoop()	573
6.214.2.2 UnregisterFromPlayerLoop()	573
6.214.3 Member Data Documentation	574
6.214.3.1 RenderSettings	574
6.214.3.2 UpdateSettings	574
6.215 NetworkRunnerUpdaterDefault.NetworkRunnerRender Struct Reference	574
6.215.1 Detailed Description	574
6.216 NetworkRunnerUpdaterDefault.NetworkRunnerUpdate Struct Reference	574
6.216.1 Detailed Description	574
6.217 NetworkRunnerUpdaterDefault.InvokeSettings Struct Reference	575
6.217.1 Detailed Description	575
6.217.2 Member Function Documentation	575
6.217.2.1 Equals() [1/2]	575

6.217.2.2 Equals() [2/2]	576
6.217.2.3 GetHashCode()	576
6.217.2.4 operator"!=()"	576
6.217.2.5 operator==()	577
6.217.2.6 ToString()	577
6.217.3 Member Data Documentation	577
6.217.3.1 AddMode	578
6.217.3.2 ReferencePlayerLoopSystem	578
6.218 NetworkSceneAsyncOp Struct Reference	578
6.218.1 Detailed Description	579
6.218.2 Member Function Documentation	579
6.218.2.1 AddOnCompleted()	579
6.218.2.2 FromAsyncOperation()	579
6.218.2.3 FromCompleted()	580
6.218.2.4 FromCoroutine()	580
6.218.2.5 FromError()	581
6.218.2.6 FromTask()	581
6.218.2.7 GetAwaiter()	582
6.218.3 Member Data Documentation	582
6.218.3.1 SceneRef	582
6.218.4 Property Documentation	582
6.218.4.1 Error	582
6.218.4.2 IsDone	582
6.218.4.3 IsValid	582
6.219 NetworkSceneAsyncOp.Awaiter Struct Reference	583
6.219.1 Detailed Description	583
6.219.2 Constructor & Destructor Documentation	583
6.219.2.1 Awaiter()	583
6.219.3 Member Function Documentation	584
6.219.3.1 GetResult()	584
6.219.3.2 OnCompleted()	584
6.219.4 Property Documentation	584
6.219.4.1 IsCompleted	584
6.220 NetworkSceneInfo Struct Reference	584
6.220.1 Detailed Description	585
6.220.2 Member Function Documentation	586
6.220.2.1 AddSceneRef()	586
6.220.2.2 Equals() [1/2]	586
6.220.2.3 Equals() [2/2]	586
6.220.2.4 GetHashCode()	587
6.220.2.5 IndexOf()	587
6.220.2.6 operator NetworkSceneInfo()	587

6.220.2.7 RemoveSceneRef()	588
6.220.2.8 ToString()	588
6.220.3 Member Data Documentation	588
6.220.3.1 MaxScenes	588
6.220.3.2 SIZE	589
6.220.3.3 WORD_COUNT	589
6.220.4 Property Documentation	589
6.220.4.1 SceneCount	589
6.220.4.2 SceneParams	589
6.220.4.3 Scenes	589
6.220.4.4 Version	589
6.221 NetworkSceneLoadId Struct Reference	590
6.221.1 Detailed Description	590
6.221.2 Constructor & Destructor Documentation	590
6.221.2.1 NetworkSceneLoadId()	590
6.221.3 Member Function Documentation	590
6.221.3.1 Equals() [1/2]	591
6.221.3.2 Equals() [2/2]	591
6.221.3.3 GetHashCode()	591
6.221.4 Member Data Documentation	591
6.221.4.1 Value	591
6.222 NetworkSceneObjectId Struct Reference	592
6.222.1 Detailed Description	592
6.222.2 Member Function Documentation	592
6.222.2.1 Equals() [1/2]	592
6.222.2.2 Equals() [2/2]	593
6.222.2.3 GetHashCode()	593
6.222.2.4 ToString()	593
6.222.3 Member Data Documentation	593
6.222.3.1 ObjectId	594
6.222.3.2 Scene	594
6.222.3.3 SceneLoadId	594
6.222.4 Property Documentation	594
6.222.4.1 IsValid	594
6.223 NetworkSerializeMethodAttribute Class Reference	594
6.223.1 Detailed Description	595
6.223.2 Property Documentation	595
6.223.2.1 MaxSize	595
6.224 NetworkSimulationConfiguration Class Reference	595
6.224.1 Detailed Description	596
6.224.2 Member Function Documentation	596
6.224.2.1 Clone()	596

6.224.2.2 Create()	596
6.224.3 Member Data Documentation	596
6.224.3.1 AdditionalJitter	597
6.224.3.2 AdditionalLoss	597
6.224.3.3 DelayMax	597
6.224.3.4 DelayMin	597
6.224.3.5 DelayPeriod	597
6.224.3.6 DelayShape	597
6.224.3.7 DelayThreshold	598
6.224.3.8 Enabled	598
6.224.3.9 LossChanceMax	598
6.224.3.10 LossChanceMin	598
6.224.3.11 LossChancePeriod	598
6.224.3.12 LossChanceShape	598
6.224.3.13 LossChanceThreshold	599
6.225 NetworkSpawnOp Struct Reference	599
6.225.1 Detailed Description	599
6.225.2 Member Function Documentation	600
6.225.2.1 GetAwaiter()	600
6.225.3 Member Data Documentation	600
6.225.3.1 Runner	600
6.225.4 Property Documentation	600
6.225.4.1 IsFailed	600
6.225.4.2 IsQueued	600
6.225.4.3 IsSpawned	600
6.225.4.4 Object	601
6.225.4.5 Status	601
6.226 NetworkSpawnOp.Awaiter Struct Reference	601
6.226.1 Detailed Description	601
6.226.2 Constructor & Destructor Documentation	601
6.226.2.1 Awaiter()	601
6.226.3 Member Function Documentation	602
6.226.3.1 GetResult()	602
6.226.3.2 OnCompleted()	602
6.226.4 Property Documentation	602
6.226.4.1 IsCompleted	603
6.227 NetworkString< TSize > Class Template Reference	603
6.227.1 Detailed Description	605
6.227.2 Constructor & Destructor Documentation	606
6.227.2.1 NetworkString()	606
6.227.3 Member Function Documentation	606
6.227.3.1 Assign()	606

6.227.3.2 Compare() [1/3]	606
6.227.3.3 Compare() [2/3]	607
6.227.3.4 Compare() [3/3]	607
6.227.3.5 Compare< TOtherSize >() [1/2]	607
6.227.3.6 Compare< TOtherSize >() [2/2]	608
6.227.3.7 Contains() [1/3]	609
6.227.3.8 Contains() [2/3]	609
6.227.3.9 Contains() [3/3]	609
6.227.3.10 Contains< TOtherSize >() [1/2]	610
6.227.3.11 Contains< TOtherSize >() [2/2]	610
6.227.3.12 EndsWith()	611
6.227.3.13 EndsWith< TOtherSize >()	611
6.227.3.14 Equals() [1/4]	612
6.227.3.15 Equals() [2/4]	612
6.227.3.16 Equals() [3/4]	613
6.227.3.17 Equals() [4/4]	613
6.227.3.18 Equals< TOtherSize >() [1/2]	613
6.227.3.19 Equals< TOtherSize >() [2/2]	614
6.227.3.20 Get()	614
6.227.3.21 GetCapacity< TSize >()	615
6.227.3.22 GetCharCount()	615
6.227.3.23 GetEnumerator()	615
6.227.3.24 GetHashCode()	616
6.227.3.25 IndexOf() [1/6]	616
6.227.3.26 IndexOf() [2/6]	616
6.227.3.27 IndexOf() [3/6]	617
6.227.3.28 IndexOf() [4/6]	617
6.227.3.29 IndexOf() [5/6]	618
6.227.3.30 IndexOf() [6/6]	618
6.227.3.31 IndexOf< TOtherSize >() [1/4]	619
6.227.3.32 IndexOf< TOtherSize >() [2/4]	619
6.227.3.33 IndexOf< TOtherSize >() [3/4]	620
6.227.3.34 IndexOf< TOtherSize >() [4/4]	621
6.227.3.35 operator NetworkString< TSize >()	622
6.227.3.36 operator string()	622
6.227.3.37 operator"!=() [1/3]	622
6.227.3.38 operator"!=() [2/3]	623
6.227.3.39 operator"!=() [3/3]	623
6.227.3.40 operator==() [1/3]	623
6.227.3.41 operator==() [2/3]	624
6.227.3.42 operator==() [3/3]	624
6.227.3.43 Set()	625

6.227.3.44 StartsWith()	625
6.227.3.45 StartsWith< TOtherSize >()	625
6.227.3.46 Substring() [1/2]	626
6.227.3.47 Substring() [2/2]	626
6.227.3.48 ToLower()	627
6.227.3.49 ToString()	627
6.227.3.50 ToUpper()	627
6.227.4 Property Documentation	628
6.227.4.1 Capacity	628
6.227.4.2 Length	628
6.227.4.3 this[int index]	628
6.227.4.4 Value	628
6.228 NetworkStructUtils Class Reference	628
6.228.1 Detailed Description	629
6.228.2 Member Function Documentation	629
6.228.2.1 GetWordCount< T >()	629
6.229 NetworkStructWeavedAttribute Class Reference	629
6.229.1 Detailed Description	630
6.229.2 Constructor & Destructor Documentation	630
6.229.2.1 NetworkStructWeavedAttribute()	630
6.229.3 Property Documentation	630
6.229.3.1 WordCount	630
6.230 NetworkTransform Class Reference	630
6.230.1 Detailed Description	631
6.230.2 Member Function Documentation	631
6.230.2.1 SetAreaOfInterestOverride()	631
6.230.2.2 Teleport()	632
6.230.3 Member Data Documentation	632
6.230.3.1 DisableSharedModelInterpolation	632
6.230.3.2 SyncParent	632
6.230.3.3 SyncScale	632
6.230.4 Property Documentation	633
6.230.4.1 AutoUpdateAreaOfInterestOverride	633
6.231 NetworkTRSP Class Reference	633
6.231.1 Detailed Description	634
6.231.2 Member Function Documentation	634
6.231.2.1 Render()	634
6.231.2.2 ResolveAOIOVERRIDE()	634
6.231.2.3 SetAreaOfInterestOverride()	635
6.231.2.4 SetParentTransform()	635
6.231.2.5 Teleport()	635
6.231.3 Property Documentation	635

6.231.3.1 Data	636
6.231.3.2 IsMainTRSP	636
6.231.3.3 State	636
6.232 NetworkTRSPData Struct Reference	636
6.232.1 Detailed Description	637
6.232.2 Member Data Documentation	637
6.232.2.1 AreaOfInterestOverride	637
6.232.2.2 Parent	637
6.232.2.3 Position	637
6.232.2.4 POSITION_OFFSET	638
6.232.2.5 Rotation	638
6.232.2.6 Scale	638
6.232.2.7 SIZE	638
6.232.2.8 TeleportKey	638
6.232.2.9 WORDS	638
6.232.3 Property Documentation	639
6.232.3.1 NonNetworkedParent	639
6.233 NormalizedRectAttribute Class Reference	639
6.233.1 Detailed Description	639
6.233.2 Constructor & Destructor Documentation	639
6.233.2.1 NormalizedRectAttribute()	639
6.233.3 Member Data Documentation	640
6.233.3.1 AspectRatio	640
6.233.3.2 InvertY	640
6.234 OnChangedRenderAttribute Class Reference	640
6.234.1 Detailed Description	640
6.234.2 Constructor & Destructor Documentation	641
6.234.2.1 OnChangedRenderAttribute()	641
6.234.3 Property Documentation	641
6.234.3.1 MethodName	641
6.235 PlayerRef Struct Reference	641
6.235.1 Detailed Description	643
6.235.2 Member Function Documentation	643
6.235.2.1 Equals() [1/2]	643
6.235.2.2 Equals() [2/2]	643
6.235.2.3 FromEncoded()	644
6.235.2.4 FromIndex()	644
6.235.2.5 GetHashCode()	644
6.235.2.6 operator"!=()"	644
6.235.2.7 operator==()	645
6.235.2.8 Read()	645
6.235.2.9 ToString()	646

6.235.2.10 Write()	646
6.235.2.11 Write< T >()	646
6.235.3 Member Data Documentation	646
6.235.3.1 MASTER_CLIENT_RAW	647
6.235.3.2 SIZE	647
6.235.4 Property Documentation	647
6.235.4.1 AsIndex	647
6.235.4.2 Comparer	647
6.235.4.3 IsMasterClient	647
6.235.4.4 IsNone	648
6.235.4.5 IsRealPlayer	648
6.235.4.6 MasterClient	648
6.235.4.7 None	648
6.235.4.8 PlayerId	648
6.235.4.9 RawEncoded	648
6.236 PreserveInPluginAttribute Class Reference	649
6.236.1 Detailed Description	649
6.236.2 Constructor & Destructor Documentation	649
6.236.2.1 PreserveInPluginAttribute()	649
6.237 IMessage Interface Reference	649
6.237.1 Detailed Description	649
6.238 Versioning Class Reference	649
6.238.1 Detailed Description	650
6.238.2 Property Documentation	650
6.238.2.1 GetCurrentVersion	650
6.238.2.2 GetProductVersion	650
6.239 Ptr Struct Reference	650
6.239.1 Detailed Description	651
6.239.2 Member Function Documentation	651
6.239.2.1 Equals() [1/2]	651
6.239.2.2 Equals() [2/2]	652
6.239.2.3 GetHashCode()	652
6.239.2.4 operator bool()	652
6.239.2.5 operator"!=()"	653
6.239.2.6 operator+()	653
6.239.2.7 operator-()	654
6.239.2.8 operator==()	654
6.239.2.9 ToString()	654
6.239.3 Member Data Documentation	655
6.239.3.1 Address	655
6.239.3.2 SIZE	655
6.239.4 Property Documentation	655

6.239.4.1 Null	655
6.240 Ptr.EqualityComparer Class Reference	655
6.240.1 Detailed Description	655
6.240.2 Member Function Documentation	656
6.240.2.1 Equals()	656
6.240.2.2 GetHashCode()	657
6.241 QuaternionCompressed Struct Reference	657
6.241.1 Detailed Description	658
6.241.2 Member Function Documentation	658
6.241.2.1 Equals() [1/2]	658
6.241.2.2 Equals() [2/2]	659
6.241.2.3 GetHashCode()	659
6.241.2.4 operator Quaternion()	659
6.241.2.5 operator QuaternionCompressed()	660
6.241.2.6 operator"!="()	660
6.241.2.7 operator==()	660
6.241.3 Member Data Documentation	661
6.241.3.1 wEncoded	661
6.241.3.2 xEncoded	661
6.241.3.3 yEncoded	661
6.241.3.4 zEncoded	661
6.241.4 Property Documentation	661
6.241.4.1 W	662
6.241.4.2 X	662
6.241.4.3 Y	662
6.241.4.4 Z	662
6.242 ReadWriteUtils Class Reference	662
6.242.1 Detailed Description	663
6.242.2 Member Function Documentation	663
6.242.2.1 ReadFloat()	663
6.242.2.2 ReadNetworkBehaviourRef()	664
6.242.2.3 ReadQuaternion()	664
6.242.2.4 ReadVector2()	665
6.242.2.5 ReadVector3()	665
6.242.2.6 ReadVector4()	665
6.242.2.7 WriteEmptyNetworkBehaviourRef()	666
6.242.2.8 WriteFloat()	666
6.242.2.9 WriteNetworkBehaviourRef()	666
6.242.2.10 WriteNullBehaviourRef()	667
6.242.2.11 WriteQuaternion()	667
6.242.2.12 WriteVector2()	667
6.242.2.13 WriteVector3()	667

6.242.2.14 WriteVector4()	669
6.242.3 Member Data Documentation	669
6.242.3.1 ACCURACY	669
6.243 ReadWriteUtilsForWeaver Class Reference	669
6.243.1 Detailed Description	670
6.243.2 Member Function Documentation	670
6.243.2.1 GetByteArrayHashCode()	670
6.243.2.2 GetByteCountUtf8NoHash()	671
6.243.2.3 GetStringHashCode()	671
6.243.2.4 GetWordCountString()	671
6.243.2.5 ReadBoolean()	672
6.243.2.6 ReadStringUtf32NoHash()	672
6.243.2.7 ReadStringUtf32WithHash()	672
6.243.2.8 ReadStringUtf8NoHash()	674
6.243.2.9 VerifyRawNetworkUnwrap< T >()	674
6.243.2.10 VerifyRawNetworkWrap< T >()	675
6.243.2.11 WriteBoolean()	675
6.243.2.12 WriteStringUtf32NoHash()	676
6.243.2.13 WriteStringUtf32WithHash()	676
6.243.2.14 WriteStringUtf8NoHash()	676
6.244 ReflectionUtils Class Reference	677
6.244.1 Detailed Description	677
6.244.2 Member Function Documentation	677
6.244.2.1 GetAllNetworkBehaviourTypes()	678
6.244.2.2 GetAllSimulationBehaviourTypes()	678
6.244.2.3 GetAllWeavedAssemblies()	678
6.244.2.4 GetAllWeavedNetworkBehaviourTypes()	678
6.244.2.5 GetAllWeavedSimulationBehaviourTypes()	679
6.244.2.6 GetAllWeaverGeneratedTypes()	679
6.244.2.7 GetCustomAttributeOrThrow< T >()	679
6.244.2.8 GetWeavedAttributeOrThrow()	680
6.245 RenderAttribute Class Reference	680
6.245.1 Detailed Description	681
6.245.2 Constructor & Destructor Documentation	681
6.245.2.1 RenderAttribute() [1/2]	681
6.245.2.2 RenderAttribute() [2/2]	681
6.245.3 Property Documentation	681
6.245.3.1 Method	681
6.245.3.2 Source	681
6.245.3.3 Timeframe	682
6.246 RenderTimeline Struct Reference	682
6.246.1 Detailed Description	682

6.246.2 Member Function Documentation	682
6.246.2.1 GetRenderBuffers()	682
6.247 RenderWeavedAttribute Class Reference	683
6.247.1 Detailed Description	683
6.247.2 Constructor & Destructor Documentation	683
6.247.2.1 RenderWeavedAttribute()	683
6.248 ResolveNetworkPrefabSourceAttribute Class Reference	683
6.248.1 Detailed Description	683
6.249 RpcAttribute Class Reference	683
6.249.1 Detailed Description	684
6.249.2 Constructor & Destructor Documentation	685
6.249.2.1 RpcAttribute() [1/2]	685
6.249.2.2 RpcAttribute() [2/2]	685
6.249.3 Member Data Documentation	685
6.249.3.1 MaxPayloadSize	685
6.249.4 Property Documentation	685
6.249.4.1 Channel	685
6.249.4.2 HostMode	686
6.249.4.3 InvokeLocal	686
6.249.4.4 Sources	686
6.249.4.5 Targets	686
6.249.4.6 TickAligned	686
6.250 RpcHeader Struct Reference	686
6.250.1 Detailed Description	687
6.250.2 Member Function Documentation	687
6.250.2.1 Create() [1/2]	687
6.250.2.2 Create() [2/2]	688
6.250.2.3 Read()	688
6.250.2.4 ReadSize()	689
6.250.2.5 ToString()	689
6.250.2.6 Write()	689
6.250.3 Member Data Documentation	689
6.250.3.1 Behaviour	690
6.250.3.2 Method	690
6.250.3.3 Object	690
6.250.3.4 SIZE	690
6.251 RpclInfo Struct Reference	690
6.251.1 Detailed Description	691
6.251.2 Member Function Documentation	691
6.251.2.1 FromLocal()	691
6.251.2.2 FromMessage()	691
6.251.2.3 ToString()	692

6.251.3 Member Data Documentation	692
6.251.3.1 Channel	692
6.251.3.2 IsInvokeLocal	692
6.251.3.3 Source	692
6.251.3.4 Tick	693
6.252 RpcInvokeData Struct Reference	693
6.252.1 Detailed Description	693
6.252.2 Member Function Documentation	693
6.252.2.1 ToString()	693
6.252.3 Member Data Documentation	694
6.252.3.1 Delegate	694
6.252.3.2 Key	694
6.252.3.3 Sources	694
6.252.3.4 Targets	694
6.253 RpcInvokeInfo Struct Reference	694
6.253.1 Detailed Description	695
6.253.2 Member Function Documentation	695
6.253.2.1 ToString()	695
6.253.3 Member Data Documentation	695
6.253.3.1 LocalInvokeResult	695
6.253.3.2 SendCullResult	695
6.253.3.3 SendResult	696
6.254 RpcSendResult Struct Reference	696
6.254.1 Detailed Description	696
6.254.2 Member Function Documentation	696
6.254.2.1 ToString()	696
6.254.3 Member Data Documentation	696
6.254.3.1 MessageSize	697
6.254.3.2 Result	697
6.255 RpcTargetAttribute Class Reference	697
6.255.1 Detailed Description	697
6.255.2 Constructor & Destructor Documentation	697
6.255.2.1 RpcTargetAttribute()	697
6.256 SceneLoadDoneArgs Struct Reference	698
6.256.1 Detailed Description	698
6.256.2 Constructor & Destructor Documentation	698
6.256.2.1 SceneLoadDoneArgs()	698
6.256.3 Member Data Documentation	699
6.256.3.1 RootGameObjects	699
6.256.3.2 Scene	699
6.256.3.3 SceneObjects	699
6.256.3.4 SceneRef	699

6.257 SceneRef Struct Reference	699
6.257.1 Detailed Description	701
6.257.2 Member Function Documentation	701
6.257.2.1 Equals() [1/2]	701
6.257.2.2 Equals() [2/2]	701
6.257.2.3 FromIndex()	702
6.257.2.4 FromPath()	702
6.257.2.5 FromRaw()	703
6.257.2.6 GetHashCode()	703
6.257.2.7 IsPath()	703
6.257.2.8 operator"!=()"	704
6.257.2.9 operator==()	704
6.257.2.10 ToString() [1/2]	704
6.257.2.11 ToString() [2/2]	705
6.257.3 Member Data Documentation	705
6.257.3.1 FLAG_ADDRESSABLE	705
6.257.3.2 RawValue	705
6.257.3.3 SIZE	705
6.257.4 Property Documentation	706
6.257.4.1 AsIndex	706
6.257.4.2 AsPathHash	706
6.257.4.3 IsIndex	706
6.257.4.4 IsValid	706
6.257.4.5 None	706
6.258 SerializableDictionary< TKey, TValue > Class Template Reference	707
6.258.1 Detailed Description	708
6.258.2 Member Function Documentation	709
6.258.2.1 Add()	709
6.258.2.2 Clear()	709
6.258.2.3 ContainsKey()	709
6.258.2.4 Create< TKey, TValue >()	710
6.258.2.5 GetEnumerator()	710
6.258.2.6 Remove()	710
6.258.2.7 Reset()	711
6.258.2.8 Store()	711
6.258.2.9 TryGetValue()	711
6.258.2.10 Wrap()	711
6.258.3 Member Data Documentation	712
6.258.3.1 EntryKeyPropertyPath	712
6.258.3.2 ItemsPropertyPath	712
6.258.4 Property Documentation	712
6.258.4.1 Count	712

6.258.4.2 IsReadOnly	712
6.258.4.3 Keys	712
6.258.4.4 this[TKey key]	712
6.258.4.5 Values	713
6.259 SessionInfo Class Reference	713
6.259.1 Detailed Description	714
6.259.2 Member Function Documentation	714
6.259.2.1 operator bool()	714
6.259.2.2 ToString()	714
6.259.2.3 UpdateCustomProperties()	714
6.259.3 Property Documentation	715
6.259.3.1 IsOpen	715
6.259.3.2 IsValid	715
6.259.3.3 IsVisible	715
6.259.3.4 MaxPlayers	715
6.259.3.5 Name	715
6.259.3.6 PlayerCount	716
6.259.3.7 Properties	716
6.259.3.8 Region	716
6.260 Simulation Class Reference	716
6.260.1 Detailed Description	719
6.260.2 Member Function Documentation	720
6.260.2.1 AfterSimulation()	720
6.260.2.2 AfterUpdate()	720
6.260.2.3 BeforeFirstTick()	720
6.260.2.4 BeforeSimulation()	720
6.260.2.5 BeforeUpdate()	720
6.260.2.6 GetAreaOfInterestGizmoData()	720
6.260.2.7 GetInputAuthority()	721
6.260.2.8 GetInputForPlayer()	721
6.260.2.9 GetObjectsAndPlayersInAreaOfInterestCell()	721
6.260.2.10 GetStateAuthority()	722
6.260.2.11 HasAnyActiveConnections()	722
6.260.2.12 IsInputAuthority()	722
6.260.2.13 IsInterestedIn()	723
6.260.2.14 IsLocalSimulationInputAuthority()	723
6.260.2.15 IsLocalSimulationStateAuthority() [1/2]	723
6.260.2.16 IsLocalSimulationStateAuthority() [2/2]	724
6.260.2.17 IsLocalSimulationStateOrInputSource()	724
6.260.2.18 IsStateAuthority() [1/2]	724
6.260.2.19 IsStateAuthority() [2/2]	725
6.260.2.20 NetworkConnected()	725

6.260.2.21 NetworkDisconnected()	725
6.260.2.22 NetworkReceiveDone()	726
6.260.2.23 NoSimulation()	726
6.260.2.24 TryGetHostPlayer()	726
6.260.2.25 Update()	726
6.260.3 Property Documentation	727
6.260.3.1 ActivePlayers	727
6.260.3.2 Config	727
6.260.3.3 DeltaTime	727
6.260.3.4 InputCount	727
6.260.3.5 IsClient	728
6.260.3.6 IsFirstTick	728
6.260.3.7 IsForward	728
6.260.3.8 IsLastTick	728
6.260.3.9 IsLocalPlayerFirstExecution	728
6.260.3.10 IsMasterClient	729
6.260.3.11 IsPlayer	729
6.260.3.12 IsResimulation	729
6.260.3.13 IsRunning	729
6.260.3.14 IsServer	729
6.260.3.15 IsShutdown	729
6.260.3.16 IsSinglePlayer	730
6.260.3.17 LatestServerTick	730
6.260.3.18 LocalAddress	730
6.260.3.19 LocalAlpha	730
6.260.3.20 LocalPlayer	730
6.260.3.21 Mode	730
6.260.3.22 NetConfigPointer	731
6.260.3.23 ObjectCount	731
6.260.3.24 Objects	731
6.260.3.25 ProjectConfig	731
6.260.3.26 RemoteAlpha	731
6.260.3.27 RemoteTick	731
6.260.3.28 RemoteTickPrevious	732
6.260.3.29 SendDelta	732
6.260.3.30 SendRate	732
6.260.3.31 Stage	732
6.260.3.32 Tick	732
6.260.3.33 TickDeltaDouble	732
6.260.3.34 TickDeltaFloat	733
6.260.3.35 TickPrevious	733
6.260.3.36 TickRate	733

6.260.3.37 TickStride	733
6.260.3.38 Time	733
6.260.3.39 Topology	733
6.261 Simulation.AreaOfInterest Struct Reference	734
6.261.1 Detailed Description	734
6.261.2 Member Function Documentation	734
6.261.2.1 GetCellSize()	734
6.261.2.2 SphereToCells()	735
6.261.2.3 ToCell() [1/2]	736
6.261.2.4 ToCell() [2/2]	736
6.261.2.5 ToCellCenter()	737
6.261.3 Member Data Documentation	737
6.261.3.1 CELL_SIZE	737
6.261.3.2 x	737
6.262 SimulationBehaviour Class Reference	738
6.262.1 Detailed Description	739
6.262.2 Member Function Documentation	739
6.262.2.1 FixedUpdateNetwork()	739
6.262.2.2 Render()	739
6.262.3 Property Documentation	739
6.262.3.1 CanReceiveRenderCallback	739
6.262.3.2 CanReceiveSimulationCallback	740
6.262.3.3 Object	740
6.262.3.4 Runner	740
6.263 SimulationBehaviourAttribute Class Reference	740
6.263.1 Detailed Description	740
6.263.2 Property Documentation	741
6.263.2.1 Modes	741
6.263.2.2 Stages	741
6.263.2.3 Topologies	741
6.264 SimulationBehaviourListScope Struct Reference	741
6.264.1 Detailed Description	741
6.264.2 Member Function Documentation	742
6.264.2.1 Dispose()	742
6.265 SimulationConfig Class Reference	742
6.265.1 Detailed Description	743
6.265.2 Member Enumeration Documentation	743
6.265.2.1 DataConsistency	743
6.265.2.2 InputTransferModes	743
6.265.3 Member Data Documentation	743
6.265.3.1 HostMigration	743
6.265.3.2 InputDataWordCount	744

6.265.3.3 InputTransferMode	744
6.265.3.4 ObjectDataConsistency	744
6.265.3.5 PlayerCount	744
6.265.3.6 ReplicationFeatures	744
6.265.3.7 TickRateSelection	744
6.265.3.8 Topology	745
6.265.4 Property Documentation	745
6.265.4.1 AreaOfInterestEnabled	745
6.265.4.2 InputTotalWordCount	745
6.265.4.3 SchedulingEnabled	745
6.265.4.4 SchedulingWithoutAOI	745
6.266 SimulationInput Class Reference	745
6.266.1 Detailed Description	746
6.266.2 Member Function Documentation	746
6.266.2.1 Clear()	746
6.266.2.2 CopyFrom()	746
6.266.3 Property Documentation	747
6.266.3.1 Data	747
6.266.3.2 Header	747
6.266.3.3 Player	747
6.266.3.4 Sent	747
6.267 SimulationInput.Buffer Class Reference	747
6.267.1 Detailed Description	748
6.267.2 Constructor & Destructor Documentation	748
6.267.2.1 Buffer()	748
6.267.3 Member Function Documentation	749
6.267.3.1 Add()	749
6.267.3.2 Clear()	749
6.267.3.3 Contains()	749
6.267.3.4 CopySortedTo()	750
6.267.3.5 Get()	750
6.267.3.6 GetInsertTime()	750
6.267.3.7 GetLastUsedInputHeader()	751
6.267.3.8 Remove()	751
6.267.4 Property Documentation	751
6.267.4.1 Count	751
6.267.4.2 Full	751
6.268 SimulationInputHeader Struct Reference	752
6.268.1 Detailed Description	752
6.268.2 Member Data Documentation	752
6.268.2.1 InterpAlpha	752
6.268.2.2 InterpFrom	752

6.268.2.3 InterpTo	753
6.268.2.4 SIZE	753
6.268.2.5 Tick	753
6.268.2.6 WORD_COUNT	753
6.269 SimulationMessage Struct Reference	753
6.269.1 Detailed Description	756
6.269.2 Member Function Documentation	756
6.269.2.1 Allocate()	756
6.269.2.2 CanAllocateUserPayload()	756
6.269.2.3 Clone()	756
6.269.2.4 GetData()	757
6.269.2.5 GetFlag()	757
6.269.2.6 IsTargeted()	757
6.269.2.7 ReadInt()	758
6.269.2.8 ReadNetworkedObjectRef()	758
6.269.2.9 ReadVector3()	758
6.269.2.10 ReferenceCountAdd()	759
6.269.2.11 ReferenceCountSub()	759
6.269.2.12 SetDummy()	759
6.269.2.13 SetNotTickAligned()	759
6.269.2.14 SetStatic()	759
6.269.2.15 SetTarget()	759
6.269.2.16 SetUnreliable()	760
6.269.2.17 ToString() [1/2]	760
6.269.2.18 ToString() [2/2]	760
6.269.2.19 WriteInt()	760
6.269.2.20 WriteNetworkedObjectRef()	761
6.269.2.21 WriteVector3()	761
6.269.3 Member Data Documentation	761
6.269.3.1 Capacity	761
6.269.3.2 FLAG_DUMMY	761
6.269.3.3 FLAG_INTERNAL	762
6.269.3.4 FLAG_NOT_TICK_ALIGNED	762
6.269.3.5 FLAG_REMOTE	762
6.269.3.6 FLAG_STATIC	762
6.269.3.7 FLAG_TARGET_PLAYER	762
6.269.3.8 FLAG_TARGET_SERVER	762
6.269.3.9 FLAG_UNRELIABLE	763
6.269.3.10 FLAG_USER_FLAGS_START	763
6.269.3.11 FLAG_USER_MESSAGE	763
6.269.3.12 Flags	763
6.269.3.13 FLAGS_RESERVED	763

6.269.3.14 FLAGS_RESERVED_BITS	763
6.269.3.15 MAX_PAYLOAD_SIZE	764
6.269.3.16 Offset	764
6.269.3.17 References	764
6.269.3.18 SIZE	764
6.269.3.19 Source	764
6.269.3.20 Target	764
6.269.3.21 Tick	765
6.269.4 Property Documentation	765
6.269.4.1 IsUnreliable	765
6.270 SimulationMessagePtr Struct Reference	765
6.270.1 Detailed Description	765
6.270.2 Member Data Documentation	765
6.270.2.1 Message	765
6.271 SimulationRuntimeConfig Struct Reference	765
6.271.1 Detailed Description	766
6.271.2 Member Data Documentation	766
6.271.2.1 HostPlayer	766
6.271.2.2 MasterClient	766
6.271.2.3 PlayerMaxCount	766
6.271.2.4 ServerMode	767
6.271.2.5 TickRate	767
6.271.2.6 Topology	767
6.272 NetAddress Struct Reference	767
6.272.1 Detailed Description	768
6.272.2 Member Function Documentation	768
6.272.2.1 Any()	768
6.272.2.2 AnyIPv6()	769
6.272.2.3 CreateFromIpPort()	769
6.272.2.4 FromActorId()	769
6.272.2.5 LocalhostIPv4()	770
6.272.2.6 LocalhostIPv6()	770
6.272.3 Property Documentation	770
6.272.3.1 ActorId	771
6.272.3.2 HasAddress	771
6.272.3.3 IsIPv4	771
6.272.3.4 IsIPv6	771
6.272.3.5 IsRelayAddr	771
6.272.3.6 IsValid	771
6.273 NetBitBufferList Struct Reference	772
6.273.1 Detailed Description	772
6.273.2 Member Function Documentation	772

6.273.2.1 AddFirst()	772
6.273.2.2 AddLast()	773
6.273.2.3 IsInList()	773
6.273.2.4 Remove()	773
6.273.2.5 RemoveHead()	773
6.274 NetCommandAccepted Struct Reference	774
6.274.1 Detailed Description	774
6.275 NetCommandConnect Struct Reference	774
6.275.1 Detailed Description	775
6.276 NetCommandDisconnect Struct Reference	775
6.276.1 Detailed Description	775
6.277 NetCommandHeader Struct Reference	775
6.277.1 Detailed Description	776
6.277.2 Member Function Documentation	776
6.277.2.1 Create()	776
6.278 NetCommandRefused Struct Reference	776
6.278.1 Detailed Description	777
6.279 NetConfig Struct Reference	777
6.279.1 Detailed Description	778
6.279.2 Member Data Documentation	778
6.279.2.1 Address	778
6.279.2.2 ConnectAttempts	778
6.279.2.3 ConnectInterval	778
6.279.2.4 ConnectionDefaultRtt	778
6.279.2.5 ConnectionGroups	779
6.279.2.6 ConnectionPingInterval	779
6.279.2.7 ConnectionSendBuffers	779
6.279.2.8 ConnectionShutdownTime	779
6.279.2.9 ConnectionTimeout	779
6.279.2.10 DefaultMtu	779
6.279.2.11 MaxConnections	780
6.279.2.12 Notify	780
6.279.2.13 OperationExpireTime	780
6.279.2.14 PacketSize	780
6.279.2.15 Simulation	780
6.279.2.16 SocketRecvBuffer	780
6.279.2.17 SocketSendBuffer	781
6.279.3 Property Documentation	781
6.279.3.1 ConnectionsPerGroup	781
6.279.3.2 Defaults	781
6.279.3.3 PacketSizeInBits	781
6.280 StunServers.StunServer Class Reference	781

6.280.1 Detailed Description	782
6.281 StartGameArgs Struct Reference	782
6.281.1 Detailed Description	783
6.281.2 Member Function Documentation	783
6.281.2.1 ToString()	784
6.281.3 Member Data Documentation	784
6.281.3.1 Address	784
6.281.3.2 AuthValues	784
6.281.3.3 Config	784
6.281.3.4 ConnectionToken	784
6.281.3.5 CustomCallbackInterfaces	785
6.281.3.6 CustomLobbyName	785
6.281.3.7 CustomPhotonAppSettings	785
6.281.3.8 CustomPublicAddress	785
6.281.3.9 CustomSTUNServer	785
6.281.3.10 DisableNATPunchthrough	785
6.281.3.11 EnableClientSessionCreation	786
6.281.3.12 GameMode	786
6.281.3.13 HostMigrationResume	786
6.281.3.14 HostMigrationToken	786
6.281.3.15 IsOpen	786
6.281.3.16 IsVisible	786
6.281.3.17 MatchmakingMode	787
6.281.3.18 ObjectInitializer	787
6.281.3.19 ObjectProvider	787
6.281.3.20 OnGameStarted	787
6.281.3.21 PlayerCount	787
6.281.3.22 Scene	787
6.281.3.23 SceneManager	788
6.281.3.24 SessionName	788
6.281.3.25 SessionNameGenerator	788
6.281.3.26 SessionProperties	788
6.281.3.27 StartGameCancellationToken	788
6.281.3.28 Updater	788
6.281.3.29 UseCachedRegions	789
6.281.3.30 UseDefaultPhotonCloudPorts	789
6.282 StartGameResult Class Reference	789
6.282.1 Detailed Description	789
6.282.2 Member Function Documentation	789
6.282.2.1 ToString()	790
6.282.3 Property Documentation	790
6.282.3.1 ErrorMessage	790

6.282.3.2 Ok	790
6.282.3.3 ShutdownReason	790
6.282.3.4 StackTrace	790
6.283 FusionStatisticsManager Class Reference	790
6.283.1 Detailed Description	791
6.283.2 Property Documentation	791
6.283.2.1 CompleteSnapshot	791
6.283.2.2 ObjectStatisticsManager	791
6.284 FusionStatisticsSnapshot Class Reference	791
6.284.1 Detailed Description	792
6.284.2 Property Documentation	792
6.284.2.1 ForwardTicks	793
6.284.2.2 GeneralAllocMemoryUsedInBytes	793
6.284.2.3 InBandwidth	793
6.284.2.4 InObjectUpdates	793
6.284.2.5 InPackets	793
6.284.2.6 InputInBandwidth	793
6.284.2.7 InputOutBandwidth	794
6.284.2.8 InputReceiveDelta	794
6.284.2.9 InterpolationOffset	794
6.284.2.10 InterpolationSpeed	794
6.284.2.11 ObjectsAllocMemoryUsedInBytes	794
6.284.2.12 OutBandwidth	794
6.284.2.13 OutObjectUpdates	795
6.284.2.14 OutPackets	795
6.284.2.15 Resimulations	795
6.284.2.16 RoundTripTime	795
6.284.2.17 SimulationSpeed	795
6.284.2.18 SimulationTimeOffset	795
6.284.2.19 StateReceiveDelta	796
6.284.2.20 TimeResets	796
6.285 NetworkObjectStatisticsManager Class Reference	796
6.285.1 Detailed Description	796
6.285.2 Member Function Documentation	796
6.285.2.1 ClearMonitoredNetworkObjects()	796
6.285.2.2 GetNetworkObjectStatistics()	797
6.285.2.3 MonitorNetworkObjectStatistics()	797
6.286 NetworkObjectStatisticsSnapshot Class Reference	797
6.286.1 Detailed Description	797
6.286.2 Property Documentation	798
6.286.2.1 InBandwidth	798
6.286.2.2 InPackets	798

6.286.2.3 OutBandwidth	798
6.286.2.4 OutPackets	798
6.287 Tick Struct Reference	798
6.287.1 Detailed Description	800
6.287.2 Member Function Documentation	800
6.287.2.1 CompareTo()	800
6.287.2.2 Equals() [1/2]	800
6.287.2.3 Equals() [2/2]	800
6.287.2.4 GetHashCode()	801
6.287.2.5 Next()	801
6.287.2.6 operator bool()	801
6.287.2.7 operator int()	802
6.287.2.8 operator Tick()	802
6.287.2.9 operator"!="()	802
6.287.2.10 operator<()	803
6.287.2.11 operator<=()	803
6.287.2.12 operator==()	803
6.287.2.13 operator>()	803
6.287.2.14 operator>=()	803
6.287.2.15 ToString()	804
6.287.3 Member Data Documentation	804
6.287.3.1 ALIGNMENT	804
6.287.3.2 Raw	804
6.287.3.3 SIZE	804
6.288 Tick.EqualityComparer Class Reference	804
6.288.1 Detailed Description	805
6.288.2 Member Function Documentation	805
6.288.2.1 Equals()	805
6.288.2.2 GetHashCode()	805
6.289 Tick.RelationalComparer Class Reference	805
6.289.1 Detailed Description	806
6.289.2 Member Function Documentation	806
6.289.2.1 Compare()	806
6.290 TickAccumulator Struct Reference	806
6.290.1 Detailed Description	807
6.290.2 Member Function Documentation	807
6.290.2.1 AddTicks()	807
6.290.2.2 AddTime()	808
6.290.2.3 Alpha()	808
6.290.2.4 ConsumeTick()	808
6.290.2.5 Start()	809
6.290.2.6 StartNew()	809

6.290.2.7 Stop()	809
6.290.3 Property Documentation	809
6.290.3.1 Pending	809
6.290.3.2 Remainder	809
6.290.3.3 Running	809
6.290.3.4 TimeScale	809
6.291 TickRate Struct Reference	810
6.291.1 Detailed Description	811
6.291.2 Member Enumeration Documentation	811
6.291.2.1 ValidateResult	811
6.291.3 Member Function Documentation	812
6.291.3.1 ClampSelection()	812
6.291.3.2 Get()	812
6.291.3.3 GetDivisor()	812
6.291.3.4 GetTickRate()	813
6.291.3.5 Init()	813
6.291.3.6 IsValid() [1/2]	814
6.291.3.7 IsValid() [2/2]	814
6.291.3.8 Resolve()	814
6.291.3.9 ToArray()	815
6.291.3.10 Validate()	815
6.291.3.11 ValidateSelection()	815
6.291.4 Property Documentation	816
6.291.4.1 Available	816
6.291.4.2 Client	816
6.291.4.3 Count	816
6.291.4.4 this[int index]	816
6.292 TickRate.Resolved Struct Reference	817
6.292.1 Detailed Description	818
6.292.2 Member Data Documentation	818
6.292.2.1 Client	818
6.292.2.2 ClientSend	818
6.292.2.3 Server	818
6.292.2.4 ServerSend	818
6.292.2.5 SIZE	818
6.292.2.6 WORDS	819
6.292.3 Property Documentation	819
6.292.3.1 ClientSendDelta	819
6.292.3.2 ClientTickDelta	819
6.292.3.3 ClientTickStride	819
6.292.3.4 ServerSendDelta	819
6.292.3.5 ServerTickDelta	819

6.292.3.6 ServerTickStride	820
6.293 TickRate.Selection Struct Reference	820
6.293.1 Detailed Description	820
6.293.2 Member Data Documentation	820
6.293.2.1 Client	820
6.293.2.2 ClientSendIndex	820
6.293.2.3 ServerIndex	821
6.293.2.4 ServerSendIndex	821
6.294 TickTimer Struct Reference	821
6.294.1 Detailed Description	822
6.294.2 Member Function Documentation	822
6.294.2.1 CreateFromSeconds()	822
6.294.2.2 CreateFromTicks()	822
6.294.2.3 Expired()	823
6.294.2.4 ExpiredOrNotRunning()	823
6.294.2.5 RemainingTicks()	824
6.294.2.6 RemainingTime()	824
6.294.2.7 ToString()	824
6.294.3 Property Documentation	824
6.294.3.1 IsRunning	825
6.294.3.2 None	825
6.294.3.3 TargetTick	825
6.295 TimeSyncConfiguration Class Reference	825
6.295.1 Detailed Description	825
6.295.2 Member Data Documentation	825
6.295.2.1 MaxLateInputs	826
6.295.2.2 MaxLateSnapshots	826
6.295.2.3 RedundantInputs	826
6.295.2.4 RedundantSnapshots	826
6.295.2.5 SampleWindowSeconds	826
6.296 UnitAttribute Class Reference	826
6.296.1 Detailed Description	827
6.297 UnityContextMenuItemAttribute Class Reference	827
6.297.1 Detailed Description	827
6.297.2 Constructor & Destructor Documentation	827
6.297.2.1 UnityContextMenuItemAttribute()	827
6.297.3 Property Documentation	828
6.297.3.1 order	828
6.298 UnityDelayedAttribute Class Reference	828
6.298.1 Detailed Description	828
6.298.2 Property Documentation	828
6.298.2.1 order	828

6.299 UnityFormerlySerializedAsAttribute Class Reference	828
6.299.1 Detailed Description	829
6.299.2 Constructor & Destructor Documentation	829
6.299.2.1 UnityFormerlySerializedAsAttribute()	829
6.300 UnityHeaderAttribute Class Reference	829
6.300.1 Detailed Description	829
6.300.2 Constructor & Destructor Documentation	829
6.300.2.1 UnityHeaderAttribute()	830
6.300.3 Property Documentation	830
6.300.3.1 order	830
6.301 UnityMinAttribute Class Reference	830
6.301.1 Detailed Description	830
6.301.2 Constructor & Destructor Documentation	830
6.301.2.1 UnityMinAttribute()	831
6.301.3 Property Documentation	831
6.301.3.1 order	831
6.302 UnityMultilineAttribute Class Reference	831
6.302.1 Detailed Description	831
6.302.2 Property Documentation	831
6.302.2.1 order	831
6.303 UnityNonReorderableAttribute Class Reference	832
6.303.1 Detailed Description	832
6.303.2 Property Documentation	832
6.303.2.1 order	832
6.304 UnityNonSerializedAttribute Class Reference	832
6.304.1 Detailed Description	832
6.305 UnityRangeAttribute Class Reference	832
6.305.1 Detailed Description	833
6.305.2 Constructor & Destructor Documentation	833
6.305.2.1 UnityRangeAttribute()	833
6.305.3 Property Documentation	833
6.305.3.1 order	833
6.306 UnitySerializeField Class Reference	833
6.306.1 Detailed Description	834
6.307 UnitySerializeReference Class Reference	834
6.307.1 Detailed Description	834
6.308 UnitySpaceAttribute Class Reference	834
6.308.1 Detailed Description	834
6.308.2 Constructor & Destructor Documentation	834
6.308.2.1 UnitySpaceAttribute()	834
6.308.3 Property Documentation	835
6.308.3.1 order	835

6.309 UnityTooltipAttribute Class Reference	835
6.309.1 Detailed Description	835
6.309.2 Constructor & Destructor Documentation	835
6.309.2.1 UnityTooltipAttribute()	835
6.309.3 Property Documentation	835
6.309.3.1 order	836
6.310 Vector2Compressed Struct Reference	836
6.310.1 Detailed Description	837
6.310.2 Member Function Documentation	837
6.310.2.1 Equals() [1/2]	837
6.310.2.2 Equals() [2/2]	837
6.310.2.3 GetHashCode()	837
6.310.2.4 operator Vector2()	838
6.310.2.5 operator Vector2Compressed()	838
6.310.2.6 operator"!="()	838
6.310.2.7 operator==()	839
6.310.3 Member Data Documentation	839
6.310.3.1 xEncoded	839
6.310.3.2 yEncoded	839
6.310.4 Property Documentation	840
6.310.4.1 X	840
6.310.4.2 Y	840
6.311 Vector3Compressed Struct Reference	840
6.311.1 Detailed Description	841
6.311.2 Member Function Documentation	841
6.311.2.1 Equals() [1/2]	841
6.311.2.2 Equals() [2/2]	842
6.311.2.3 GetHashCode()	842
6.311.2.4 operator Vector2()	842
6.311.2.5 operator Vector3()	843
6.311.2.6 operator Vector3Compressed() [1/2]	843
6.311.2.7 operator Vector3Compressed() [2/2]	843
6.311.2.8 operator"!="()	844
6.311.2.9 operator==()	844
6.311.3 Member Data Documentation	844
6.311.3.1 xEncoded	844
6.311.3.2 yEncoded	845
6.311.3.3 zEncoded	845
6.311.4 Property Documentation	845
6.311.4.1 X	845
6.311.4.2 Y	845
6.311.4.3 Z	845

6.312 Vector4Compressed Struct Reference	845
6.312.1 Detailed Description	846
6.312.2 Member Function Documentation	847
6.312.2.1 Equals() [1/2]	847
6.312.2.2 Equals() [2/2]	847
6.312.2.3 GetHashCode()	847
6.312.2.4 operator Vector4()	848
6.312.2.5 operator Vector4Compressed()	848
6.312.2.6 operator"!="()	848
6.312.2.7 operator==()	849
6.312.3 Member Data Documentation	849
6.312.3.1 wEncoded	849
6.312.3.2 xEncoded	849
6.312.3.3 yEncoded	850
6.312.3.4 zEncoded	850
6.312.4 Property Documentation	850
6.312.4.1 W	850
6.312.4.2 X	850
6.312.4.3 Y	850
6.312.4.4 Z	850
6.313 WarnIfAttribute Class Reference	851
6.313.1 Detailed Description	851
6.313.2 Member Data Documentation	851
6.313.2.1 Message	851
6.314 WeaverGeneratedAttribute Class Reference	851
6.314.1 Detailed Description	851
Index	853

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.

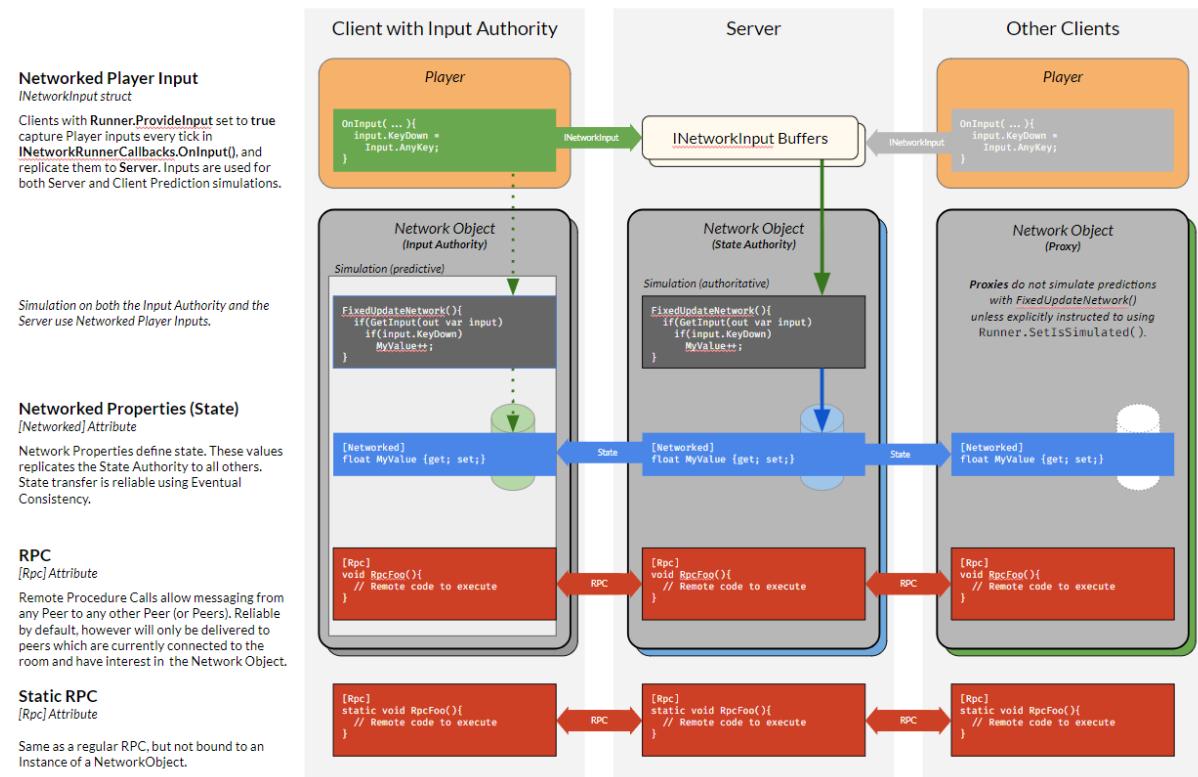


Figure 2.1 Overview of the Core Fusion APIs

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 chose 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

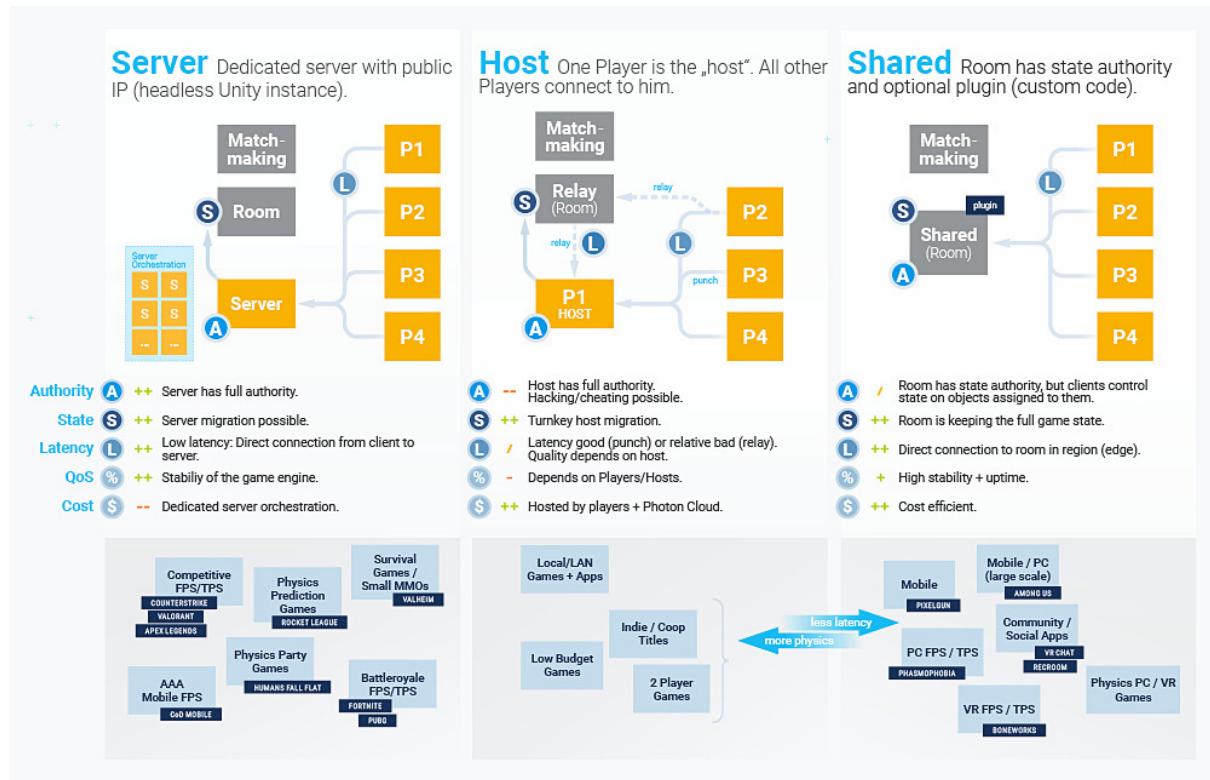


Figure 2.3 Fusion Network Topologies

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	69
Allocator.Config	71
AssetObject	84
TaskManager	85
AuthorityMasks	87
Behaviour	88
Hitbox	128
NetworkEvents	370
NetworkObject	408
NetworkObjectPrefabData	445
NetworkRunner	509
SimulationBehaviour	738
HitboxManager	132
NetworkBehaviour	266
HitboxRoot	146
NetworkMecanimAnimator	405
NetworkTRSP	633
NetworkTransform	630
CapacityAttribute	90
DefaultForPropertyAttribute	91
DisplayAsEnumAttribute	92
DoflAttributeBase	93
DrawIfAttribute	93
WarnIfAttribute	851
DynamicHeap	95
DynamicHeap.Ignore	99
DynamicHeapInstance	99
IDataEncryption	105
DataEncryptor	104
FieldsMask< T >	108
FixedSize< T >	111
FixedSize< T >.Enumerator	117
FixedBufferPropertyAttribute	119
FixedStorage	121

FloatUtils	125
HeapConfiguration	127
HostMigrationConfig	151
HostMigrationToken	152
IAfterAllTicks	152
NetworkMecanimAnimator	405
NetworkTransform	630
IAfterClientPredictionReset	153
IAfterHostMigration	154
IAfterRender	154
IAfterSpawned	155
IAfterTick	155
HitboxManager	132
IAfterUpdate	156
IAsyncOperation	157
ICoroutine	162
IBeforeAllTicks	158
NetworkTransform	630
IBeforeClientPredictionReset	159
IBeforeCopyPreviousState	159
NetworkTransform	630
IBeforeHitboxRegistration	160
IBeforeSimulation	160
HitboxManager	132
IBeforeTick	161
IBeforeUpdate	162
IDespawned	163
NetworkBehaviour	266
IElementReaderWriter< T >	163
IFixedStorage	167
_128	61
_16	62
_2	63
_256	63
_32	64
_4	65
_512	66
_64	67
_8	68
IInputAuthorityGained	167
IInputAuthorityLost	168
IInterestEnter	168
IInterestExit	169
ILocalPrefabCreated	169
INetworkArray	170
NetworkArray< T >	255
INetworkAssetSource< T >	171
INetworkDictionary	172
NetworkDictionary< K, V >	358
INetworkInput	173
INetworkLinkedList	173
NetworkLinkedList< T >	388
INetworkObjectInitializer	174
NetworkObjectInitializerUnity	439
INetworkObjectProvider	175

NetworkObjectProviderDummy	446
INetworkRunnerCallbacks	177
NetworkDelegates	357
NetworkEvents	370
INetworkRunnerUpdater	183
NetworkRunnerUpdaterDefault	572
INetworkSceneManager	185
INetworkStruct	189
Angle	75
FloatCompressed	122
NetworkBehaviourId	318
NetworkBool	340
NetworkButtons	343
NetworkId	376
NetworkObjectGuid	422
NetworkObjectHeader	430
NetworkObjectNestingKey	440
NetworkObjectTypeid	450
NetworkPhysicsInfo	461
NetworkPrefabId	464
NetworkPrefabRef	472
NetworkRNG	502
NetworkSceneInfo	584
NetworkString< TSize >	603
NetworkTRSPData	636
PlayerRef	641
Ptr	650
QuaternionCompressed	657
SceneRef	699
TickTimer	821
Vector2Compressed	836
Vector3Compressed	840
Vector4Compressed	845
_128	61
_16	62
_2	63
_256	63
_32	64
_4	65
_512	66
_64	67
_8	68
INetworkTRSPTeleport	189
NetworkTransform	630
IUnitySurrogate	190
IUnityValueSurrogate< T >	191
UnityValueSurrogate< T, TReaderWriter >	200
UnitySurrogateBase	198
UnityArraySurrogate< T, ReaderWriter >	192
UnityDictionarySurrogate< TKeyType, TKeyReaderWriter, TValueType, TValueReaderWriter >	194
UnityLinkedListSurrogate< T, ReaderWriter >	196
UnityValueSurrogate< T, TReaderWriter >	200
InterpolatedErrorCorrectionSettings	202
IPlayerJoined	205
IPlayerLeft	205
IRemotePrefabCreated	206
ISceneLoadDone	207

ISceneLoadStart	207
ISimulationEnter	208
ISimulationExit	209
ISpawned	209
HitboxManager	132
NetworkBehaviour	266
IStateAuthorityChanged	210
LagCompensatedHit	210
AABB	214
BoxOverlapQueryParams	218
BVHDraw	220
BVHNodeDrawInfo	221
ColliderDrawInfo	222
HitboxColliderContainerDraw	224
LagCompensatedExt	224
LagCompensationDraw	225
LagCompensationUtils.ContactData	226
PositionRotationQueryParams	227
Query	229
BoxOverlapQuery	216
RaycastQuery	236
RaycastAllQuery	235
SphereOverlapQuery	241
QueryParams	233
RaycastQueryParams	238
SnapshotHistoryDraw	240
SphereOverlapQueryParams	243
LagCompensationSettings	244
LobbyInfo	246
LogSimpleUnity	247
NestedComponentUtilities	247
NetworkArray< T >.Enumerator	262
NetworkArrayExtensions	264
NetworkArrayReadOnly< T >	265
NetworkAssemblyIgnoreAttribute	266
NetworkAssemblyWeavedAttribute	266
NetworkBehaviour.ArrayReader< T >	289
NetworkBehaviour.BehaviourReader< T >	290
NetworkBehaviour.ChangeDetector	291
NetworkBehaviour.ChangeDetector.Enumerable	294
NetworkBehaviour.ChangeDetector.Enumerator	295
NetworkBehaviour.DictionaryReader< K, V >	296
NetworkBehaviour.LinkListReader< T >	297
NetworkBehaviour.PropertyReader< T >	298
NetworkBehaviourBuffer	299
NetworkBehaviourBufferInterpolator	307
NetworkBehaviourUtils	322
NetworkBehaviourUtils.ArrayInitializer< T >	336
NetworkBehaviourUtils.DictionaryInitializer< K, V >	338
NetworkBehaviourUtils.MetaData	339
NetworkBehaviourWeavedAttribute	339
NetworkConfiguration	353
NetworkDeserializeMethodAttribute	357
NetworkDictionary< K, V >.Enumerator	364
NetworkDictionaryReadOnly< K, V >	366
NetworkedAttribute	368
NetworkedWeavedAttribute	369
NetworkEvents.ConnectFailedEvent	372

NetworkEvents.ConnectRequestEvent	372
NetworkEvents.CustomAuthenticationResponse	372
NetworkEvents.DisconnectFromServerEvent	372
NetworkEvents.HostMigrationEvent	373
NetworkEvents.InputEvent	373
NetworkEvents.InputPlayerEvent	373
NetworkEvents.ObjectEvent	373
NetworkEvents.ObjectPlayerEvent	374
NetworkEvents.PlayerEvent	374
NetworkEvents.ReliableDataEvent	374
NetworkEvents.ReliableProgressEvent	374
NetworkEvents.RunnerEvent	375
NetworkEvents.SessionListUpdateEvent	375
NetworkEvents.ShutdownEvent	375
NetworkEvents.SimulationMessageEvent	375
NetworkId.EqualityComparer	382
NetworkInput	382
NetworkInputUtils	386
NetworkInputWeavedAttribute	387
NetworkLinkedList< T >.Enumerator	395
NetworkLinkedListReadOnly< T >	397
NetworkLoadSceneParameters	401
NetworkObjectFlagsExtensions	419
NetworkObjectGuid.EqualityComparer	430
NetworkObjectHeaderPtr	437
NetworkObjectMeta	439
NetworkObjectNestingKey.EqualityComparer	444
NetworkObjectReleaseContext	446
NetworkObjectSortKeyComparer	448
NetworkObjectSpawnException	449
NetworkObjectTypeid.EqualityComparer	459
NetworkPrefabAcquireContext	462
NetworkPrefabAttribute	464
NetworkPrefabId.EqualityComparer	469
NetworkPrefabInfo	470
NetworkPrefabRef.EqualityComparer	479
NetworkPrefabTable	480
NetworkPrefabTableOptions	488
NetworkProjectConfig	489
NetworkProjectConfigAsset	498
NetworkProjectConfigAsset.SerializableSimulationBehaviourMeta	501
NetworkRpcStaticWeavedInvokerAttribute	507
NetworkRpcWeavedInvokerAttribute	508
NetworkRunnerCallbackArgs	570
NetworkRunnerCallbackArgs.ConnectRequest	571
NetworkRunnerUpdaterDefault.NetworkRunnerRender	574
NetworkRunnerUpdaterDefault.NetworkRunnerUpdate	574
NetworkRunnerUpdaterDefaultInvokeSettings	575
NetworkSceneAsyncOp	578
NetworkSceneAsyncOp.Awaiter	583
NetworkSceneLoadId	590
NetworkSceneObjectId	592
NetworkSerializeMethodAttribute	594
NetworkSimulationConfiguration	595
NetworkSpawnOp	599
NetworkSpawnOp.Awaiter	601
NetworkStructUtils	628
NetworkStructWeavedAttribute	629

NormalizedRectAttribute	639
OnChangedRenderAttribute	640
PreserveInPluginAttribute	649
IMessage	649
Versioning	649
Ptr.EqualityComparer	655
ReadWriteUtils	662
ReadWriteUtilsForWeaver	669
ReflectionUtils	677
RenderAttribute	680
RenderTimeline	682
RenderWeavedAttribute	683
ResolveNetworkPrefabSourceAttribute	683
RpcAttribute	683
RpcHeader	686
RpcInfo	690
RpcInvokeData	693
RpcInvokeInfo	694
RpcSendResult	696
RpcTargetAttribute	697
SceneLoadDoneArgs	698
SerializableDictionary< TKey, TValue >	707
SessionInfo	713
Simulation	716
Simulation.AreaOfInterest	734
SimulationBehaviourAttribute	740
SimulationBehaviourListScope	741
SimulationConfig	742
SimulationInput	745
SimulationInput.Buffer	747
SimulationInputHeader	752
SimulationMessage	753
SimulationMessagePtr	765
SimulationRuntimeConfig	765
NetAddress	767
NetBitBufferList	772
NetCommandAccepted	774
NetCommandConnect	774
NetCommandDisconnect	775
NetCommandHeader	775
NetCommandRefused	776
NetConfig	777
StunServers.StunServer	781
StartGameArgs	782
StartGameResult	789
FusionStatisticsManager	790
FusionStatisticsSnapshot	791
NetworkObjectStatisticsManager	796
NetworkObjectStatisticsSnapshot	797
Tick	798
Tick.EqualityComparer	804
Tick.RelationalComparer	805
TickAccumulator	806
TickRate	810
TickRate.Resolved	817
TickRate.Selection	820
TimeSyncConfiguration	825
UnitAttribute	826

UnityContextMenuMenuItemAttribute	827
UnityDelayedAttribute	828
UnityFormerlySerializedAsAttribute	828
UnityHeaderAttribute	829
UnityMinAttribute	830
UnityMultilineAttribute	831
UnityNonReorderableAttribute	832
UnityNonSerializedAttribute	832
UnityRangeAttribute	832
UnitySerializeField	833
UnitySerializeReference	834
UnitySpaceAttribute	834
UnityTooltipAttribute	835
WeaverGeneratedAttribute	851
IElementReaderWriter< K >	163
IElementReaderWriter< V >	163
INetworkAssetSource< NetworkObject >	171
INetworkPrefabSource	176
NetworkBehaviour.PropertyReader< Fusion.NetworkBehaviourId >	298

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	61
_16	A FixedStorage that can hold up to 16 words	62
_2	A FixedStorage that can hold up to 2 words	63
_256	A FixedStorage that can hold up to 256 words	63
_32	A FixedStorage that can hold up to 32 words	64
_4	A FixedStorage that can hold up to 4 words	65
_512	A FixedStorage that can hold up to 512 words	66
_64	A FixedStorage that can hold up to 64 words	67
_8	A FixedStorage that can hold up to 8 words	68
Allocator	Memory Allocator	69
Allocator.Config	Memory Allocator Configuration	71
Angle	A Networked fusion type for degrees. This can be used with the NetworkedAttribute , in RPCs, or in NetworkInput structs	75
AssetObject	Base class for all Fusion assets	84
TaskManager	Task Factory is used to create new Tasks and Schedule long running Tasks	85
AuthorityMasks	Provides constants and methods for managing authority masks	87
Behaviour	Alternative base class to Unity's MonoBehaviour. This allows for components that work both in Unity, as well as the Photon relays	88
CapacityAttribute	Capacity Attribute	90

DefaultForPropertyAttribute	
Default For Property Attribute	91
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	92
DolfAttributeBase	
Editor attribute for selective editor rendering. Condition member can be a property, field or method (with a return value)	93
DrawIfAttribute	
Editor attribute for selectively drawing/hiding fields. Condition member can be a property, field or method (with a return value)	93
DynamicHeap	
A dynamic heap for allocating and tracking unmanaged objects	95
DynamicHeap.Ignore	
Ignore this field when scanning for pointers	99
DynamicHeapInstance	
Dynamic heap instance	99
DataEncryptor	
Responsible for encrypting and decrypting data buffers	104
IDataEncryption	
Interface for classes that manage the encryption/decryption of byte arrays	105
FieldsMask< T >	
Base class for FieldsMask<T>	108
FixedArray< T >	
A fixed size array that can be used in structs	111
FixedArray< T >.Enumerator	
Enumerator for the FixedArray struct	117
FixedBufferPropertyAttribute	
Fixed Buffer Property Attribute	119
FixedStorage	
Provides utility methods for fixed storage types	121
FloatCompressed	
Represents a compressed float value for network transmission	122
FloatUtils	
Provides utility methods for compressing and decompressing float values	125
HeapConfiguration	
Memory Heap Settings	127
Hitbox	
Represents a single lag-compensated collider. Multiple component instances can be added anywhere in the hierarchy of a NetworkObject which includes a HitboxRoot	128
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	132
HitboxRoot	
Root Hitbox group container. Manages registering/unregistering hitboxes with the group, and defines the broadphase geometry for the group	146
HostMigrationConfig	
Project configuration settings specific to how the Host Migration behaves	151
HostMigrationToken	
Transitory Holder with all necessary information to restart the Fusion Runner after the Host Migration has completed	152
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	152

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	153
IAfterHostMigration	Used to mark NetworkBehaviors that need to be react after a Host Migration process	154
IAfterRender	Interface for AfterRender callback. Called after the render loop	154
IAfterSpawned	Interface for AfterSpawned callback. Called after the object is spawned	155
IAfterTick	Interface for AfterTick callback. Called after each tick simulation completes. Implement this interface on SimulationBehaviour and NetworkBehaviour classes	155
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	156
IAsyncOperation	Defines an asynchronous operation	157
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	158
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	159
IBeforeCopyPreviousState	Interface for BeforeCopyPreviousState callback. Called before the copy of the previous state	159
IBeforeHitboxRegistration	Interface for BeforeHitboxRegistration callback. Implement this interface on SimulationBehaviour and NetworkBehaviour classes	160
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	160
IBeforeTick	Interface for BeforeTick callback. Called before each tick is simulated. Implement this interface on SimulationBehaviour and NetworkBehaviour classes	161
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	162
ICoroutine	Defines a coroutine	162
IDespawned	Interface for Despawned callback. Called when a NetworkBehaviour is despawned	163
IElementReaderWriter< T >	Defines the interface for reading and writing elements in a byte array	163
IFixedStorage	Interface for fixed storage types	167
IInputAuthorityGained	Interface for handling the event when the input authority is gained	167
IInputAuthorityLost	Interface for handling the event when the input authority is lost	168
IInterestEnter	Interface for handling the event when a player enters the area of interest	168
IInterestExit	Interface for handling the event when a player exits the area of interest	169

ILocalPrefabCreated	Interface for handling the event when a local prefab is created	169
INetworkArray	Defines the interface for a networked array	170
INetworkAssetSource< T >	Interface for a network asset source	171
INetworkDictionary	Defines the interface for a networked dictionary	172
INetworkInput	Flag interface for custom NetworkInput structs	173
INetworkLinkedList	Defines the interface for a networked linked list	173
INetworkObjectInitializer	Interface for initializing network objects	174
INetworkObjectProvider	Interface which defines the handlers for NetworkRunner <code>Spawn()</code> and <code>Despawn()</code> actions. Passing an instance of this interface to NetworkRunner.StartGame(StartGameArgs) as the <code>StartGameArgs.ObjectProvider</code> argument value will assign that instance as the handler for runner <code>Spawn()</code> and <code>Despawn()</code> actions. By default (if <code>StartGameArgs.ObjectProvider == null</code>) actions will use <code>Instantiate()</code> , and <code>Despawn()</code> actions will use <code>Destroy()</code>	175
INetworkPrefabSource	Interface for a network prefab source	176
INetworkRunnerCallbacks	Interface for NetworkRunner callbacks. Register a class/struct instance which implements this interface with NetworkRunner.AddCallbacks(INetworkRunnerCallbacks[])	177
INetworkRunnerUpdater	Interface which defines the handlers for NetworkRunner Updates. An implementation is responsible for calling NetworkRunner.UpdateInternal(double) and NetworkRunner.RenderInternal periodically	183
INetworkSceneManager	Interface for a NetworkRunner scene manager. A scene manager is responsible for loading and unloading scenes	185
INetworkStruct	Base interface for all Fusion Network Structs	189
INetworkTRSPTeleport	Implement this interface on a NetworkTRSP implementation to indicate it can be teleported	189
IUnitySurrogate	Represents an interface for Unity surrogates. This interface provides methods for reading and writing data	190
IUnityValueSurrogate< T >	Represents an interface for Unity value surrogates. This interface provides a property for accessing and modifying the data	191
UnityArraySurrogate< T, ReaderWriter >	A base class for Unity array surrogates	192
UnityDictionarySurrogate< TKeyType, TKeyReaderWriter, TValueType, TValueReaderWriter >	A surrogate for serializing a dictionary	194
UnityLinkedListSurrogate< T, ReaderWriter >	A surrogate for serializing a linked list	196
UnitySurrogateBase	Represents a base class for Unity surrogates. This class is serializable and provides abstract methods for reading, writing, and initializing data	198
UnityValueSurrogate< T, TReaderWriter >	Represents a base class for Unity value surrogates. This class is serializable and provides methods for reading, writing, and initializing data	200
InterpolatedErrorCorrectionSettings	A set of parameters that tune the interpolated correction of prediction error on transform data	202
IPlayerJoined	Interface for handling the event when a player joins the game	205

IPlayerLeft	Interface for handling the event when a player leaves the game	205
IRemotePrefabCreated	Interface for handling the event when a remote prefab is created	206
ISceneLoadDone	Interface for handling the event when a scene load operation is completed	207
ISceneLoadStart	Interface for handling the event when a scene load operation is started	207
ISimulationEnter	Interface for <code>SimulationEnter</code> callback. Called when the <code>NetworkObject</code> joins <code>AreaOfInterest</code> . Implement this interface on <code>SimulationBehaviour</code> and <code>NetworkBehaviour</code> classes	208
ISimulationExit	Interface for the <code>SimulationExit</code> callback. Called when the <code>NetworkObject</code> leaves <code>AreaOfInterest</code> . Implement this interface on <code>SimulationBehaviour</code> and <code>NetworkBehaviour</code> classes	209
ISpawned	Interface for handling the event when an object is spawned	209
IStateAuthorityChanged	Interface for handling the event when the state authority changes	210
LagCompensatedHit	Defines a lag compensated query hit result	210
AABB	Represents an Axis-Aligned Bounding Box (<code>AABB</code>)	214
BoxOverlapQuery	Class that represents a box overlap query. Used to query against the <code>NetworkRunner.LagCompensation</code> API	216
BoxOverlapQueryParams	Base parameters needed to execute a box overlap query	218
BVHDraw	Provide a way to iterate over BVH and return a <code>BVHNodeDrawInfo</code> for each node	220
BVHNodeDrawInfo	Container class to provide the necessary info to draw nodes from the BVH	221
ColliderDrawInfo	Container class to provide the necessary information to draw a hitbox collider	222
HitboxColliderContainerDraw	Provide a way to iterate over the <code>HitboxBuffer.HitboxSnapshot</code> and return the <code>ColliderDrawInfo</code> for each collider on the snapshot	224
LagCompensatedExt	LagCompensated Extension methods	224
LagCompensationDraw	Provide access to iterate over the lag compensation system components and give the necessary information to draw them	225
LagCompensationUtils.ContactData	Details regarding a shape intersection. It does not carry information about the intersection happening or not	226
PositionRotationQueryParams	<code>Query</code> parameters for position rotation query	227
Query	Base class for all Lag Compensation queries	229
queryParams	Base parameters needed to execute a query	233
RaycastAllQuery	Class that represents a raycast all query. Used to query against the <code>NetworkRunner.LagCompensation</code> API	235
RaycastQuery	Class that represents a raycast query. Used to query against the <code>NetworkRunner.LagCompensation</code> API	236
RaycastQueryParams	Base parameters needed to execute a raycast query	238

SnapshotHistoryDraw	Provide a way to iterate over the HitboxBuffer and return the HitboxColliderContainerDraw container for each snapshot on the buffer	240
SphereOverlapQuery	Class that represents a sphere overlap query. Used to query against the NetworkRunner.LagCompensation API	241
SphereOverlapQueryParams	Base parameters needed to execute a sphere overlap query	243
LagCompensationSettings	Settings for lag compensation history	244
LobbyInfo	Holds information about a Lobby	246
LogSimpleUnity	Log Simple Unity	247
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	247
NetworkArray< T >	Fusion type for networking arrays. Maximum capacity is fixed, and is set with the CapacityAttribute . 255	
NetworkArray< T >.Enumerator	Enumerator for NetworkArray	262
NetworkArrayExtensions	Provides extension methods for the NetworkArray class	264
NetworkArrayReadOnly< T >	Provides a read-only view of a network array	265
NetworkAssemblyIgnoreAttribute	Network Assembly Ignore Attribute	266
NetworkAssemblyWeavedAttribute	Network Assembly Weaved Attribute	266
NetworkBehaviour	Base class for Fusion network components, which are associated with a NetworkObject	266
NetworkBehaviour.ArrayReader< T >	Provides a reader for network arrays of type T	289
NetworkBehaviour.BehaviourReader< T >	Provides a reader for network behaviours of type T	290
NetworkBehaviour.ChangeDetector	Change detector for a NetworkBehaviour	291
NetworkBehaviour.ChangeDetector.Enumerable	Struct representing a collection of changes detected in a NetworkBehaviour	294
NetworkBehaviour.ChangeDetector.Enumerator	Enumerator for the collection of changes detected in a NetworkBehaviour	295
NetworkBehaviour.DictionaryReader< K, V >	Provides a reader for network dictionaries with keys of type K and values of type V	296
NetworkBehaviour.LinkListReader< T >	Provides a reader for network linked lists of type T	297
NetworkBehaviour.PropertyReader< T >	Provides a reader for properties of type T in a network behaviour	298
NetworkBehaviourBuffer	Provides low level accesss to data buffers that can be read using a NetworkBehaviour.Reader	299
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	307

NetworkBehaviourId	Represents the unique identifier for a NetworkBehaviour instance	318
NetworkBehaviourUtils	This static class provides utility methods for working with NetworkBehaviour objects	322
NetworkBehaviourUtils.ArrayInitializer< T >	A utility structure for initializing NetworkArray and NetworkLinkedList with inline initialization	336
NetworkBehaviourUtils.DictionaryInitializer< K, V >	A utility structure for initializing NetworkDictionary with inline initialization	338
NetworkBehaviourUtils.MetaData	This structure holds metadata for a NetworkBehaviour object	339
NetworkBehaviourWeavedAttribute	Network Behaviour Weaved Attribute	339
NetworkBool	Represents a boolean value that can be networked	340
NetworkButtons	Represents a set of buttons that can be networked	343
NetworkConfiguration	Main network configuration class	353
NetworkDelegates	Network Runner Callbacks Delegates	357
NetworkDeserializeMethodAttribute	Network Deserialize Method Attribute	357
NetworkDictionary< K, V >	Fusion type for networking Dictionaries. Maximum capacity is fixed, and is set with the CapacityAttribute. 358	358
NetworkDictionary< K, V >.Enumerator	Enumerator for NetworkDictionary	364
NetworkDictionaryReadOnly< K, V >	A read-only version of NetworkDictionary< TKey, TValue >	366
NetworkedAttribute	368	368
NetworkedWeavedAttribute	Networked Weaved Attribute	369
NetworkEvents	Companion component for NetworkRunner . Exposes INetworkRunnerCallbacks as UnityEvents, which can be wired up to other components in the inspector	370
NetworkEvents.ConnectFailedEvent	UnityEvent for ConnectFailed	372
NetworkEvents.ConnectRequestEvent	UnityEvent for ConnectRequest	372
NetworkEvents.CustomAuthenticationResponse	UnityEvent for Custom Authentication	372
NetworkEvents.DisconnectFromServerEvent	UnityEvent for DisconnectFromServer	372
NetworkEvents.HostMigrationEvent	UnityEvent for HostMigration	373
NetworkEvents.InputEvent	UnityEvent for NetworkInput	373
NetworkEvents.InputPlayerEvent	UnityEvent for NetworkInput with PlayerRef	373
NetworkEvents.ObjectEvent	UnityEvent for NetworkObject	373
NetworkEvents.ObjectPlayerEvent	UnityEvent for NetworkObject with PlayerRef	374
NetworkEvents.PlayerEvent	UnityEvent for PlayerRef	374

NetworkEvents.ReliableDataEvent	UnityEvent for Reliable Data	374
NetworkEvents.ReliableProgressEvent	UnityEvent for Reliable Data Progress	374
NetworkEvents.RunnerEvent	UnityEvent for NetworkRunner	375
NetworkEvents.SessionListUpdateEvent	UnityEvent for SessionInfo List	375
NetworkEvents.ShutdownEvent	UnityEvent for Shutdown	375
NetworkEvents.SimulationMessageEvent	UnityEvent for SimulationMessage	375
NetworkId	The unique identifier for a network entity	376
NetworkId.EqualityComparer	IEqualityComparer interface for NetworkId objects	382
NetworkInput	NetworkInput Struct	382
NetworkInputUtils	Utility methods for NetworkInput	386
NetworkInputWeavedAttribute	Network Input Weaved Attribute	387
NetworkLinkedList< T >	Fusion type for networking LinkedLists. Maximum capacity is fixed, and is set with the CapacityAttribute. Typical Usage:	388
NetworkLinkedList< T >.Enumerator	Enumerator for NetworkLinkedList< T >	395
NetworkLinkedListReadOnly< T >	Read-only version of NetworkLinkedList< T >	397
NetworkLoadSceneParameters	Parameters for loading a scene	401
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	405
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	408
NetworkObjectFlagsExtensions	Extension methods for the NetworkObjectFlags enum	419
NetworkObjectGuid	NetworkObjectGuid	422
NetworkObjectGuid.EqualityComparer	EqualityComparer for NetworkObjectGuid	430
NetworkObjectHeader	Network object header information for a NetworkObject	430
NetworkObjectHeaderPtr	Represents a pointer to a NetworkObjectHeader . This struct is unsafe because it uses pointers	437
NetworkObjectInitializerUnity	Initializes network objects for Unity	439
NetworkObjectMeta	Meta information about a network object	439
NetworkObjectNestingKey	A key used to identify a network object nesting	440
NetworkObjectNestingKey.EqualityComparer	Implements the IEqualityComparer interface	444

NetworkObjectPrefabData	This class represents the data for a network object prefab	445
NetworkObjectProviderDummy	A dummy implementation of the INetworkObjectProvider interface. This class is used for testing purposes and throws a NotImplementedException for all its methods	446
NetworkObjectReleaseContext	Represents the context for releasing a network object. This struct is unsafe because it uses pointers	446
NetworkObjectSortKeyComparer	This class is used to compare two NetworkObject instances based on their SortKey. It implements the IComparer interface	448
NetworkObjectSpawnException	Network Object Spawn Exception	449
NetworkObjectTypeId	ID for a NetworkObject Prefab which has been cataloged in a NetworkProjectConfig.PrefabTable	450
NetworkObjectTypeId.EqualityComparer	NetworkObjectTypeId Comparer	459
NetworkPhysicsInfo	Network Physics INetworkStruct	461
NetworkPrefabAcquireContext	Represents the context for acquiring a prefab instance for a network object. This struct is unsafe because it uses pointers	462
NetworkPrefabAttribute	Network Prefab Attribute	464
NetworkPrefabId	ID for a NetworkObject Prefab which has been cataloged in a NetworkProjectConfig.PrefabTable	464
NetworkPrefabId.EqualityComparer	Equality comparer for NetworkPrefabId	469
NetworkPrefabInfo	Meta data for a NetworkObject prefab which has been cataloged in a NetworkProjectConfig.PrefabTable	470
NetworkPrefabRef	NetworkPrefabRef	472
NetworkPrefabRef.EqualityComparer	EqualityComparer for NetworkPrefabRef	479
NetworkPrefabTable	Class representing a table of network prefabs	480
NetworkPrefabTableOptions	Options for the NetworkPrefabTable	488
NetworkProjectConfig	The core Fusion config file that is shared with all peers at startup	489
NetworkProjectConfigAsset	Manages and references the current instance of NetworkProjectConfig	498
NetworkProjectConfigAsset.SerializableSimulationBehaviourMeta	An auto-generated list containing meta information about all the SimulationBehaviours in the project, e.g. execution order	501
NetworkRNG	PCG32 random generator, 16 bytes in size. http://www.pcg-random.org	502
NetworkRpcStaticWeavedInvokerAttribute	Network Rpc Static Weaved Invoker Attribute Contains info about a static weaved RPC Method	507
NetworkRpcWeavedInvokerAttribute	Network Rpc Weaved Invoker Attribute Contains info about a weaved RPC Method	508
NetworkRunner	Host Migration related code in order to get a copy of the Simulation State	509
NetworkRunnerCallbackArgs	Stores data types used on the INetworkRunnerCallbacks interface	570
NetworkRunnerCallbackArgs.ConnectRequest	Data holder of a Connection Request from a remote client	571

NetworkRunnerUpdaterDefault	Default implementation of INetworkRunnerUpdater that uses the Unity PlayerLoop	572
NetworkRunnerUpdaterDefault.NetworkRunnerRender	Used to invoke NetworkRunner.RenderInternal in the PlayerLoop	574
NetworkRunnerUpdaterDefault.NetworkRunnerUpdate	Used to invoke NetworkRunner.UpdateInternal(double) in the PlayerLoop	574
NetworkRunnerUpdaterDefaultInvokeSettings	Settings for the NetworkRunnerUpdaterDefault	575
NetworkSceneAsyncOp	A wrapper for async scene operations	578
NetworkSceneAsyncOp.Awaiter	Awaiter for NetworkSceneAsyncOp	583
NetworkScenelInfo	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	584
NetworkSceneLoadId	A unique identifier for a scene load operation	590
NetworkSceneObjectid	A unique identifier for a scene object	592
NetworkSerializeMethodAttribute	Network Serialize Method Attribute	594
NetworkSimulationConfiguration	Configuration for network conditions simulation (induced latency and loss)	595
NetworkSpawnOp	Spawn Operation	599
NetworkSpawnOp.Awaiter	Awaiter for NetworkSpawnOp	601
NetworkString< TSize >	Fixed-size UTF32 string. All operations are alloc-free, except for converting to System.String	603
NetworkStructUtils	Utility methods for INetworkStruct	628
NetworkStructWeavedAttribute	Describes the total number of WORDs a INetworkStruct uses	629
NetworkTransform	Add to any NetworkObject Transform, or its associated child Transforms to automatically synchronize TRSP (Position/Rotation/Scale/Parent)	630
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	633
NetworkTRSPData	Data structure storing spatial (Position/Rotation/Scale/Parent) synchronization data for spatial synchronization components, NetworkTRSP and its subclass NetworkTransform	636
NormalizedRectAttribute	Enables a special inspector drawer for Unity Rect type, specially designed for editing Rect↔ Transforms using normalized values	639
OnChangedRenderAttribute	OnChangedRender Attribute	640
PlayerRef	Represents a Fusion player	641
PreserveInPluginAttribute	Preserve In Plugin Attribute	649
IMessage	Represents a Protocol Message	649
Versioning	Versioning Information	649
Ptr	Ptr	650

Ptr.EqualityComparer	
Ptr Equality Comparer	655
QuaternionCompressed	
Represents a compressed Quaternion value for network transmission	657
ReadWriteUtils	
Provides utility methods for reading and writing data	662
ReadWriteUtilsForWeaver	
Provides utility methods for reading and writing data	669
ReflectionUtils	
Provides utility methods for reflection	677
RenderAttribute	
Override default render settings for [Networked] properties	680
RenderTimeline	
Can be used to acquire interpolated data for different points in time	682
RenderWeavedAttribute	
Render Weaved Attribute	683
ResolveNetworkPrefabSourceAttribute	
Resolve Network Prefab Source Attribute	683
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	683
RpcHeader	
Header for RPC messages	686
RpcInfo	
RpcInfo is a struct that contains information about the RPC message	690
RpcInvokeData	
Represents the data required to invoke an RPC message	693
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	694
RpcSendResult	
RPC send operation result information	696
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:	697
SceneLoadDoneArgs	
Struct that contains information about a scene after it has been loaded	698
SceneRef	
Scene reference struct. Can be used to reference a scene by index or by path	699
SerializableDictionary< TKey, TValue >	
A serializable dictionary	707
SessionInfo	
Holds information about the Game Session	713
Simulation	
Main simulation class	716
Simulation.AreaOfInterest	
Area of Interest Definition	734
SimulationBehaviour	
Base class for a Fusion aware Behaviour (derived from <code>UnityEngine.MonoBehaviour</code>). Objects derived from this object can be associated with a NetworkRunner and Simulation . If a parent NetworkObject is found, this component will also be associated with that network entity	738
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:	740

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	741
SimulationConfig	Project configuration settings specific to how the Simulation class behaves	742
SimulationInput	Simulation Input	745
SimulationInput.Buffer	Buffer for SimulationInputs	747
SimulationInputHeader	Simulation Input Header	752
SimulationMessage	Simulation Message	753
SimulationMessagePtr	Simulation Message Pointer	765
SimulationRuntimeConfig	Stores the runtime configuration of the simulation	765
NetAddress	Represents a Network Address, which includes a IP and Port This can contains either a IPv4 or a IPv6 address	767
NetBitBufferList	Represents a linked list of Fusion.Sockets.NetBitBuffer	772
NetCommandAccepted	Accepted Command, sent by the server when a remote client connection is accepted	774
NetCommandConnect	Connect Command used to signal a remote server that a client is trying to connect to it	774
NetCommandDisconnect	Disconnect Command, it can be used by either side of the connection	775
NetCommandHeader	Network Command Header Describe its type and usual settings for all commands	775
NetCommandRefused	Refuse Command, sent by the server when the connection was refused. This happens when the server has reached its max connection capacity	776
NetConfig	General configuration used to drive the behavior of the Socket library	777
StunServers.StunServer	Stores Addresses of a STUN Server	781
StartGameArgs	Fusion Start Arguments, used to configure the simulation mode and other settings	782
StartGameResult	Represents the result of starting the Fusion Simulation	789
FusionStatisticsManager	Represents a fusion statistics manager	790
FusionStatisticsSnapshot	Represents a snapshot of Fusion statistics	791
NetworkObjectStatisticsManager	Manages network object statistics for monitored network objects	796
NetworkObjectStatisticsSnapshot	Represents a snapshot of network object statistics	797
Tick	A tick is a 32-bit integer that represents a frame number	798
Tick.EqualityComparer	Provides a mechanism for comparing two Tick objects for equality	804
Tick.RelationalComparer	Provides a mechanism for comparing two Tick objects	805
TickAccumulator	A tick accumulator	806

TickRate	A tick rate is a collection of tick rates	810
TickRate.Resolved	Represents a resolved tick rate	817
TickRate.Selection	Represents a selection of tick rates for client and server	820
TickTimer	A timer that is based on ticks instead of seconds	821
TimeSyncConfiguration	Time Synchronization Configuration	825
UnitAttribute	Unit Attribute class. Used to mark a field with the respective Units	826
UnityContextMenuAttribute	Unity ContextMenuItemAttribute	827
UnityDelayedAttribute	Unity DelayedAttribute	828
UnityFormerlySerializedAsAttribute	Unity FormerlySerializedAsAttribute	828
UnityHeaderAttribute	Unity HeaderAttribute	829
UnityMinAttribute	Unity MinAttribute	830
UnityMultilineAttribute	Unity MultilineAttribute	831
UnityNonReorderableAttribute	Unity NonReorderableAttribute	832
UnityNonSerializedAttribute	Unity NonSerializedAttribute	832
UnityRangeAttribute	Unity RangeAttribute	832
UnitySerializeField	Unity SerializeField	833
UnitySerializeReference	Unity SerializeReference	834
UnitySpaceAttribute	Unity SpaceAttribute	834
UnityTooltipAttribute	Unity TooltipAttribute	835
Vector2Compressed	Represents a compressed Vector2 value for network transmission	836
Vector3Compressed	Represents a compressed Vector3 value for network transmission	840
Vector4Compressed	Represents a compressed Vector4 value for network transmission	845
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)	851
WeaverGeneratedAttribute	Weaver Generated Attribute	851

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 [AssetObject](#)
Base class for all *Fusion* assets.
- 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 [CapacityAttribute](#)

- class [Capacity Attribute](#)
- 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 [DolfAttributeBase](#)

Editor attribute for selective editor rendering. Condition member can be a property, field or method (with a return value).
- class [DrawIfAttribute](#)

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

A dynamic heap for allocating and tracking unmanaged objects.
- class [DynamicHeapInstance](#)

Dynamic heap instance.
- 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 [HeapConfiguration](#)

Memory Heap Settings
- 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 [IAfterSpawned](#)

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

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

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

- **INetworkDictionary**
Interface for a network asset source.
- **INetworkInput**
Defines the interface for a networked dictionary.
- **INetworkInput**
Flag interface for custom NetworkInput structs.
- **INetworkLinkedList**
Defines the interface for a networked linked list.
- **INetworkObjectInitializer**
Interface for initializing network objects.
- **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().
- **INetworkPrefabSource**
Interface for a network prefab source.
- **INetworkRunnerCallbacks**
Interface for NetworkRunner callbacks. Register a class/struct instance which implements this interface with NetworkRunner.AddCallbacks(INetworkRunnerCallbacks[]).
- **INetworkRunnerUpdater**
Interface which defines the handlers for NetworkRunner Updates. An implementation is responsible for calling NetworkRunner.UpdateInternal(double) and NetworkRunner.RenderInternal periodically.
- **INetworkSceneManager**
Interface for a NetworkRunner scene manager. A scene manager is responsible for loading and unloading scenes
- **INetworkStruct**
Base interface for all Fusion Network Structs
- **INetworkTRSPTeleport**
Implement this interface on a NetworkTRSP implementation to indicate it can be teleported.
- **InterpolatedErrorCorrectionSettings**
A set of parameters that tune the interpolated correction of prediction error on transform data.
- **IPlayerJoined**
Interface for handling the event when a player joins the game.
- **IPlayerLeft**
Interface for handling the event when a player leaves the game.
- **IRemotePrefabCreated**
Interface for handling the event when a remote prefab is created.
- **ISceneLoadDone**
Interface for handling the event when a scene load operation is completed.
- **ISceneLoadStart**
Interface for handling the event when a scene load operation is started.
- **ISimulationEnter**
Interface for SimulationEnter callback. Called when the NetworkObject joins AreaOfInterest. Implement this interface on SimulationBehaviour and NetworkBehaviour classes.
- **ISimulationExit**
Interface for the SimulationExit callback. Called when the NetworkObject leaves AreaOfInterest. Implement this interface on SimulationBehaviour and NetworkBehaviour classes.
- **ISpawned**
Interface for handling the event when an object is spawned.
- **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 [LobbyInfo](#)
Holds information about a Lobby
- class [LogSimpleUnity](#)
Log Simple Unity
- 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 NetworkBehaviour.Reader
- 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 [NetworkedWeavedAttribute](#)

- Networked Weaved Attribute*
- 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](#)

- struct **NetworkObjectTypeld**

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
- struct [Ptr](#)
Ptr
- struct [QuaternionCompressed](#)
Represents a compressed Quaternion value for network transmission.
- 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:
- struct [SceneLoadDoneArgs](#)
Struct that contains information about a scene after it has been loaded.
- struct [SceneRef](#)
Scene reference struct. Can be used to reference a scene by index or by path.
- class [SerializableDictionary](#)
A serializable dictionary.
- 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`). Objects derived from this object can be associated with a [NetworkRunner](#) and [Simulation](#). If a parent [NetworkObject](#) is found, this component will also be associated with that network entity.
- 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](#)

- struct [TickRate](#)
A tick accumulator.
- struct [TickTimer](#)
A timer that is based on ticks instead of seconds.
- class [TimeSyncConfiguration](#)
Time Synchronization Configuration
- class [UnitAttribute](#)
Unit Attribute class. Used to mark a field with the respective [Units](#)
- class [UnityContextMenuItemAttribute](#)
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 [UnitySerializeField](#)
Unity SerializeField
- class [UnitySerializeReference](#)
Unity SerializeReference
- class [UnitySpaceAttribute](#)
Unity SpaceAttribute
- class [UnityTooltipAttribute](#)
Unity TooltipAttribute
- 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 [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**
- enum class **EditorButtonVisibility**
- 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 Render←Timeframe).
- 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 [RpclInvokeInfo.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](#)
- 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 FusionGlobalScriptableObjectLoadResult [FusionGlobalScriptableObjectLoadDelegate](#) (Type type)
- delegate void [FusionGlobalScriptableObjectUnloadDelegate](#) (FusionGlobalScriptableObject instance)
- delegate void [NetworkObjectSpawnDelegate](#) ([NetworkSpawnOp](#) result)
Network Object Spawn Delegate
- unsafe delegate void [RpclInvokeDelegate](#) ([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.

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 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.

Enumerator

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.

5.1.1.4 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.

5.1.1.5 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.

5.1.1.6 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.

5.1.1.7 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	Flag indicating that the network object is a master client object.
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.

5.1.1.8 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.

5.1.1.9 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.

5.1.1.10 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 remote scene has changed.

5.1.1.11 NetworkSceneInfoDefaultFlags

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

Network Scene Info Default Flags

Enumerator

SceneCountMask	The scene count mask
CounterMask	The counter mask

5.1.1.12 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 .
SharedModeStateAuthLocalPlayer	In shared mode, override the state authority to local player.

5.1.1.13 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.

5.1.1.14 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.

5.1.1.15 PageSizes

```
enum PageSizes [strong]
```

Page Bit Shift Lookup Table

5.1.1.16 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.

5.1.1.17 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.

5.1.1.18 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.

5.1.1.19 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.

5.1.1.20 RpcHostMode

enum [RpcHostMode](#) [strong]

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

Enumerator

SourcesServer	If host invokes RPC RpclInfo.Source will be set to PlayerRef.None (default).
SourcesHostPlayer	If host invokes RPC RpclInfo.Source will be set to the host's local player.

5.1.1.21 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.
TagetPlayerIsNotLocal	TargetPlayerIsNotLocal

5.1.1.22 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 RpclInvokeInfo.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.
TargetPlayerIsLocalButRpclsNotInvokableLocally	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.

5.1.1.23 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.

Enumerator

	MaskNotSent	Mask for not sent messages.
	MaskBroadcast	Mask for broadcast messages.
	MaskCulled	Mask for culled messages.

5.1.1.24 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.

5.1.1.25 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.

5.1.1.26 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.
Generated by Doxygen Remote	Represents a remote RPC target.

5.1.1.27 ScriptHeaderBackColor

```
enum ScriptHeaderBackColor [strong]
```

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

5.1.1.28 ScriptHeaderIcon

```
enum ScriptHeaderIcon [strong]
```

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

5.1.1.29 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

5.1.1.30 ShutdownReason

```
enum ShutdownReason [strong]
```

Describes a list of Reason why the [Fusion](#) Runner was Shutdown

Enumerator

Ok	OK Reason means Fusion 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.
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.

Enumerator

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
GameIsFull	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

5.1.1.31 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.

5.1.1.32 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.

5.1.1.33 Topologies

```
enum Topologies [strong]
```

Enumerator

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

5.1.1.34 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

5.1.1.35 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 NetworkObjectSpawnDelegate()

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

Network Object Spawn Delegate

5.1.2.2 RpcInvokeDelegate()

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

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

Parameters

behaviour	The NetworkBehaviour associated with the RPC message.
message	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.

5.1.2.3 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

Enumerations

- enum class **StaticFieldResetMode**

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

- interface [IMessage](#)
Represents a [Protocol](#) Message
- class [Versioning](#)
Versioning Information

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

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 occurred on the Plugin

5.8 Fusion.Runtime Namespace Reference

5.9 Fusion.Runtime.Unity Namespace Reference

5.10 Fusion.Sockets Namespace Reference

Classes

- struct [NetAddress](#)

- struct [NetBitBufferList](#)

Represents a Network Address, which includes a IP and Port This can contains either a IPv4 or a IPv6 address
- 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

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](#)
- enum class [NetDisconnectReason](#) : byte

Disconnect Reason Flag
- enum class [NetPacketType](#) : byte

Describe the type of a Networked Packet
- enum class [OnConnectionRequestReply](#)

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 NetDisconnectReason

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

Disconnect Reason Flag

5.10.1.4 NetPacketType

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

Describe the type of a Networked Packet

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.
Generated by Doxygen	
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

5.12 Fusion.Statistics Namespace Reference

Classes

- class [FusionStatisticsManager](#)
Represents a fusion statistics manager.
- class [FusionStatisticsSnapshot](#)
Represents a snapshot of [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 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.

6.12.1 Detailed Description

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

6.12.2 Member Function Documentation

6.12.2.1 Clamp() [1/2]

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

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

6.12.2.2 Clamp() [2/2]

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

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

6.12.2.3 Equals() [1/2]

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

Checks equality with another `Angle`.

Parameters

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

Returns

Equality result.

6.12.2.4 Equals() [2/2]

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

Checks equality with an object.

Parameters

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

Returns

Equality result.

6.12.2.5 GetHashCode()

```
override int GetHashCode ( )
```

Gets the hash code.

Returns

Hash code.

6.12.2.6 Lerp()

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

Lerps between two angle values.

6.12.2.7 Max()

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

Returns the larger of two supplied angles.

6.12.2.8 Min()

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

Returns the smaller of two supplied angles.

6.12.2.9 operator Angle() [1/3]

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

Converts double to [Angle](#).

Parameters

<code>value</code>	Double value.
--------------------	---------------

Returns

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

6.12.2.10 operator Angle() [2/3]

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

Converts float to [Angle](#).

Parameters

<code>value</code>	Float value.
--------------------	--------------

Returns

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

6.12.2.11 operator Angle() [3/3]

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

Converts int to [Angle](#).

Parameters

<code>value</code>	Integer value.
--------------------	----------------

Returns

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

6.12.2.12 operator double()

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

Converts [Angle](#) to double.

Parameters

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

Returns

Double representation of the [Angle](#).

6.12.2.13 operator float()

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

Converts [Angle](#) to float.

Parameters

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

Returns

Float representation of the `Angle`.

6.12.2.14 operator"!=()

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

Inequality operator for `Angle` struct.

Parameters

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

Returns

True if the value of the first `Angle` instance is not equal to the value of the second `Angle` instance, otherwise false.

6.12.2.15 operator+()

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

Addition operator for `Angle` struct.

Parameters

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

Returns

A new `Angle` instance that is the sum of the first and second `Angle` instances, wrapped around at 360 degrees if necessary.

6.12.2.16 operator-()

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

Subtraction operator for [Angle](#) struct.

Parameters

a	First Angle instance.
b	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.

6.12.2.17 operator<()

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

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

Parameters

a	First Angle instance.
b	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.

6.12.2.18 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.

6.12.2.19 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.

6.12.2.20 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.

6.12.2.21 operator>=()

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

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

Parameters

a	First Angle instance.
b	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.

6.12.2.22 ToString()

```
override string ToString ( )
```

String representation of the [Angle](#).

6.12.3 Member Data Documentation

6.12.3.1 SIZE

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

Size of this struct in bytes.

6.13 AssetObject Class Reference

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

Inherits [ScriptableObject](#).

6.13.1 Detailed Description

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

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

Run an Action asynchronously
- static Task [Service](#) (Action 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

6.14.1 Detailed Description

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

6.14.2 Member Function Documentation

6.14.2.1 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

6.14.2.2 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>	CancellationToken used to stop the Action
<i>options</i>	Extra Task Creation options

Returns

[Async](#) Task based on the Action

6.14.2.3 Service()

```
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

6.14.2.4 Setup()

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

Setup a new TaskFactory tailored to work with Unity

6.15 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.

6.15.1 Detailed Description

Provides constants and methods for managing authority masks.

6.15.2 Member Data Documentation

6.15.2.1 ALL

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

Constant representing all authorities.

6.15.2.2 INPUT

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

Constant representing the input authority mask.

6.15.2.3 NONE

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

Constant representing no authority.

6.15.2.4 PROXY

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

Constant representing the proxy authority mask.

6.15.2.5 STATE

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

Constant representing the state authority mask.

6.16 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()

6.16.1 Detailed Description

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

6.16.2 Member Function Documentation

6.16.2.1 AddBehaviour< T >()

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

Wrapper for Unity's GameObject.AddComponent()

Type Constraints

T : Behaviour

6.16.2.2 DestroyBehaviour()

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

Wrapper for Unity's GameObject.Destroy()

6.16.2.3 GetBehaviour< T >()

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

Wrapper for Unity's GameObject.GetComponentInChildren()

Type Constraints

T : Behaviour

6.16.2.4 TryGetBehaviour< T >()

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

Wrapper for Unity's GameObject.TryGetComponent()

Type Constraints

T : Behaviour

6.17 CapacityAttribute Class Reference

Capacity Attribute

Inherits Attribute.

Public Member Functions

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

Properties

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

6.17.1 Detailed Description

Capacity Attribute

6.17.2 Constructor & Destructor Documentation

6.17.2.1 CapacityAttribute()

```
CapacityAttribute (
    int length )
```

[CapacityAttribute](#) Constructor

Parameters

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

6.17.3 Property Documentation

6.17.3.1 Length

int Length [get]

Total Capacity

6.18 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

6.18.1 Detailed Description

Default For Property Attribute

For non-serialized properties

6.18.2 Constructor & Destructor Documentation

6.18.2.1 DefaultForPropertyAttribute()

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

[DefaultForPropertyAttribute](#) Constructor

Parameters

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

6.18.3 Property Documentation

6.18.3.1 PropertyName

```
String PropertyName [get]
```

Property Name

6.18.3.2 WordCount

```
int WordCount [get]
```

Property Word Count

6.18.3.3 WordOffset

```
int WordOffset [get]
```

Property Word Offset

6.19 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)
- **DisplayAsEnumAttribute** (Type enumType)

Properties

- Type **EnumType** [get]
- string **EnumTypeMemberName** [get]

6.19.1 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.

6.20 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**
- bool **_isDouble**
- long **_longValue**
- [CompareOperator](#) **Compare**
- string **ConditionMember**
- bool **ErrorOnConditionMemberNotFound** = true

Protected Member Functions

- **DolfAttributeBase** (string propertyPath, double compareToValue, [CompareOperator](#) compare)
- **DolfAttributeBase** (string propertyPath, long compareToValue, [CompareOperator](#) compare)

6.20.1 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. Null = 0, False = 0, True = 1, Unity Object = InstanceId

6.21 DrawIfAttribute Class Reference

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

Inherits [DolfAttributeBase](#).

Public Member Functions

- **DrawIfAttribute** (string propertyPath)
- **DrawIfAttribute** (string propertyPath, bool compareToValue, CompareOperator compare=CompareOperator.Equal, DrawIfMode mode=DrawIfMode.ReadOnly)

Constructor.
- **DrawIfAttribute** (string propertyPath, double compareToValue, CompareOperator compare=CompareOperator.Equal, DrawIfMode mode=DrawIfMode.ReadOnly)

Constructor.
- **DrawIfAttribute** (string propertyPath, 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]

Additional Inherited Members

6.21.1 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

6.21.2 Constructor & Destructor Documentation

6.21.2.1 DrawIfAttribute() [1/2]

```
DrawIfAttribute (
    string propertyPath,
    double compareToValue,
    CompareOperator compare = CompareOperator.Equal,
    DrawIfMode mode = DrawIfMode.ReadOnly )
```

Constructor.

Parameters

<i>propertyPath</i>	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

Parameters

<i>compareToValue</i>	The value to compare the member value against.
<i>mode</i>	How the field should be hidden (disabled or removed)
<i>compare</i>	How the condition member value and compareToValye will be evaluated.

6.21.2.2 DrawIfAttribute() [2/2]

```
DrawIfAttribute (
    string propertyPath,
    bool compareToValue,
    CompareOperator compare = CompareOperator.Equal,
    DrawIfMode mode = DrawIfMode.ReadOnly )
```

Constructor.

Parameters

<i>propertyPath</i>	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

Parameters

<i>compareToValue</i>	The value to compare the member value against.
<i>compare</i>	How the condition member value and compareToValye will be evaluated.
<i>mode</i>	How the field should be hidden (disabled or removed)

6.21.3 Member Data Documentation

6.21.3.1 Mode

DrawIfMode Mode

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

6.22 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 dynamicRootsLength)

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`

6.22.1 Detailed Description

A dynamic heap for allocating and tracking unmanaged objects.

6.22.2 Member Function Documentation

6.22.2.1 CollectGarbage()

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

Collect garbage

Parameters

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

6.22.2.2 CollectGarbageDelegate()

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

Collect garbage delegate

6.22.2.3 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
----------------------------------	--

6.22.2.4 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

6.22.3 Member Data Documentation

6.22.3.1 _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  
}
```

6.23 DynamicHeap.Ignore Class Reference

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

Inherits Attribute.

6.23.1 Detailed Description

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

6.24 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.

6.24.1 Detailed Description

Dynamic heap instance.

6.24.2 Constructor & Destructor Documentation

6.24.2.1 DynamicHeapInstance()

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

Create a dynamic heap instance.

Parameters

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

6.24.3 Member Function Documentation

6.24.3.1 Allocate()

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

Allocate a pointer.

Parameters

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

Returns

Pointer to allocated memory.

6.24.3.2 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***

6.24.3.3 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***

6.24.3.4 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*

6.24.3.5 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*

6.24.3.6 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*

6.24.3.7 Free()

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

Free a pointer.

Parameters

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

6.25 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](#) ()
- 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

6.25.1 Detailed Description

Responsible for encrypting and decrypting data buffers

6.25.2 Member Function Documentation

6.25.2.1 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](#).

6.26 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 its length.
- bool [EncryptData](#) (byte *buffer, ref int bufferLength, int capacity)
Encrypt data in place and update its 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

6.26.1 Detailed Description

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

6.26.2 Member Function Documentation

6.26.2.1 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](#).

6.26.2.2 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](#).

6.26.2.3 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](#).

6.26.2.4 GenerateKey()

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

Generate the key used used by the encryption implementation

Returns

Key used to setup the encryption implementation

Implemented in [DataEncryptor](#).

6.26.2.5 Setup()

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

Setup the encryption implementation with the right key

Implemented in [DataEncryptor](#).

6.26.2.6 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](#).

6.27 FieldsMask< T > Class Template Reference

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

Inherits FieldsMask.

Public Member Functions

- **FieldsMask ()**
Constructor for FieldsMask< T >.
- **FieldsMask (Func< Mask256 > getDefaultsDelegate)**
- **FieldsMask (long maskA, long maskB=0, long maskC=0, long maskD=0)**
- **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**

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.

6.27.1 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.

6.27.2 Constructor & Destructor Documentation

6.27.2.1 FieldsMask() [1/5]

```
FieldsMask (
    Mask256 mask )  [protected]
```

Constructor for FieldsMask.

6.27.2.2 FieldsMask() [2/5]

```
FieldsMask (
    long a,
    long b,
    long c,
    long d ) [protected]
```

Constructor for [FieldsMask](#).

6.27.2.3 FieldsMask() [3/5]

```
FieldsMask ( ) [protected]
```

Constructor for [FieldsMask](#).

6.27.2.4 FieldsMask() [4/5]

```
FieldsMask (
    Mask256 mask )
```

Constructor for [FieldsMask<T>](#).

6.27.2.5 FieldsMask() [5/5]

```
FieldsMask ( )
```

Constructor for [FieldsMask<T>](#).

6.27.3 Member Function Documentation

6.27.3.1 operator Mask256()

```
static implicit operator Mask256 (
    FieldsMask< T > mask ) [static]
```

Implicitly convert [FieldsMask](#) to its long mask value.

6.28 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.

- IEnumarator< T > IEnumerable< T >. GetEnumerator ()

- IEnumarator IEnumerable. GetEnumerator ()

- T[] ToArray ()

Allocates a new array and copies values from this array. For a non-alloc alternative use CopyTo(List< T >).

- string ToString ()

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.

6.28.1 Detailed Description

A fixed size array that can be used in structs.

Helper methods for [FixedArray](#).

Template Parameters

<code>T</code>	
----------------	--

Type Constraints

`T : unmanaged`

6.28.2 Constructor & Destructor Documentation

6.28.2.1 FixedArray()

```
FixedArray (
    T * array,
    int length )
```

[NetworkArray](#) constructor.

6.28.3 Member Function Documentation

6.28.3.1 Clear()

```
void Clear( )
```

Sets all elements in the array to their default value.

6.28.3.2 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.

6.28.3.3 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.

6.28.3.4 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.

6.28.3.5 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.

6.28.3.6 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*

6.28.3.7 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*

6.28.3.8 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**

6.28.3.9 GetEnumerator()

```
Enumerator GetEnumerator ( )
```

Returns an enumerator that iterates through the [FixedArray](#).

6.28.3.10 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*>

6.28.3.11 ToArray()

```
T [ ] ToArray ( )
```

Allocates a new array and copies values from this array. For a non-alloc alternative use [CopyTo\(List<T>\)](#).

6.28.3.12 ToString()

```
string ToString( )
```

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.

6.28.3.13 ToListString()

```
override string ToListString( )
```

Returns a string that represents the current object.

6.28.4 Property Documentation

6.28.4.1 Length

```
int Length [get]
```

The fixed size of the array.

6.28.4.2 this[int index]

```
ref T this[int index] [get]
```

Indexer of array elements.

6.29 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]`

6.29.1 Detailed Description

`Enumerator` for the `FixedArray` struct.

6.29.2 Constructor & Destructor Documentation

6.29.2.1 Enumerator()

```
Enumerator (
    FixedArray< T > array )
```

Initializes a new instance of the `Enumerator` struct.

Parameters

<code>array</code>	The <code>FixedArray</code> instance to enumerate.
--------------------	--

6.29.3 Member Function Documentation

6.29.3.1 Dispose()

```
void Dispose ( )
```

Performs application-defined tasks associated with freeing, releasing, or resetting unmanaged resources.

6.29.3.2 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.

6.29.3.3 Reset()

```
void Reset ( )
```

Sets the enumerator to its initial position, which is before the first element in the collection.

6.29.4 Property Documentation

6.29.4.1 Current

```
T Current [get]
```

Gets the current element in the collection.

6.30 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

6.30.1 Detailed Description

Fixed Buffer Property Attribute

6.30.2 Constructor & Destructor Documentation

6.30.2.1 FixedBufferPropertyAttribute()

```
FixedBufferPropertyAttribute (
    Type fieldType,
    Type surrogateType,
    int capacity )
```

FixedBufferPropertyAttribute Constructor

Parameters

<i>fieldType</i>	Type
<i>surrogateType</i>	SurrogateType
<i>capacity</i>	Capacity

6.30.3 Property Documentation

6.30.3.1 Capacity

```
int Capacity [get]
```

Fixed Buffer Capacity

6.30.3.2 SurrogateType

```
Type SurrogateType [get]
```

Fixed Buffer Surrogate Type

6.30.3.3 Type

Type Type [get]

Fixed Buffer Type

6.31 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.

6.31.1 Detailed Description

Provides utility methods for fixed storage types.

6.31.2 Member Function Documentation

6.31.2.1 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**

6.32 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.

6.32.1 Detailed Description

Represents a compressed float value for network transmission.

6.32.2 Member Function Documentation

6.32.2.1 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.

6.32.2.2 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.

6.32.2.3 GetHashCode()

```
override int GetHashCode ( )
```

Returns the hash code for the current [FloatCompressed](#) instance.

Returns

A hash code for the current [FloatCompressed](#) instance.

6.32.2.4 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.

6.32.2.5 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.

6.32.2.6 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.

6.32.2.7 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.

6.32.3 Member Data Documentation

6.32.3.1 valueEncoded

```
int valueEncoded
```

Encoded value of the float.

6.33 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.

6.33.1 Detailed Description

Provides utility methods for compressing and decompressing float values.

6.33.2 Member Function Documentation

6.33.2.1 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 DEFAULT_ACCURACY.

Returns

The compressed float value.

6.33.2.2 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 DEFAULT_ACCURACY.

Returns

The decompressed float value.

6.33.3 Member Data Documentation

6.33.3.1 DEFAULT_ACCURACY

```
const int DEFAULT_ACCURACY = 1 << 10 [static]
```

Default accuracy for float compression and decompression.

6.34 HeapConfiguration Class Reference

Memory Heap Settings

Public Member Functions

- [HeapConfiguration Init](#) (int globalsSize)
Initializes and creates a new [HeapConfiguration](#) based on the Global Size
- [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

6.34.1 Detailed Description

Memory Heap Settings

6.34.2 Member Function Documentation

6.34.2.1 Init()

```
HeapConfiguration Init (
    int globalsSize )
```

Initializes and creates a new [HeapConfiguration](#) based on the Global Size

6.34.2.2 `ToString()`

```
override string ToString ( )
```

ToString

6.34.3 Member Data Documentation

6.34.3.1 `GlobalsSize`

```
int GlobalsSize
```

Heap Global Size

6.34.3.2 `PageCount`

```
int PageCount = Allocator.Config.DEFAULT_BLOCK_COUNT
```

Default number of Heap Pages

6.34.3.3 `PageShift`

```
PageSizes PageShift = Allocator.Config.DEFAULT_BLOCK_SHIFT
```

Default size of each Heap Page

6.35 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

6.35.1 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.

6.35.2 Member Function Documentation

6.35.2.1 DrawGizmos()

```
virtual void DrawGizmos (
    Color color,
    ref Matrix4x4 localToWorldMatrix ) [protected], [virtual]
```

Draw the [Hitbox](#) gizmos.

6.35.2.2 OnDrawGizmos()

```
void OnDrawGizmos ( )
```

Draws this hitbox gizmo on Unity editor.

6.35.2.3 SetLayer()

```
void SetLayer (
    int layer )
```

Set a layer mask for this [Hitbox](#) gameobject. This method caches the layer mask.

Parameters

<i>layer</i>	<input type="text"/>
--------------	----------------------

6.35.3 Member Data Documentation

6.35.3.1 BoxExtents

```
Vector3 BoxExtents
```

When [Type](#) is set to [HitboxTypes.Box](#), this defines the local-space geometry for narrow-phase checks.

6.35.3.2 CapsuleExtents

```
float CapsuleExtents
```

When [Type](#) is set to [HitboxTypes.Capsule](#), this defines the local-space geometry for narrow-phase checks.

6.35.3.3 CapsuleRadius

```
float CapsuleRadius
```

When [Type](#) is set to [HitboxTypes.Capsule](#), this defines the local-space geometry for narrow-phase checks.

6.35.3.4 GizmosColor

```
Color GizmosColor = Color.yellow
```

Color used when drawing gizmos for this hitbox.

6.35.3.5 Offset

```
Vector3 Offset
```

This [Hitbox](#)'s local-space offset from its [GameObject](#) position.

6.35.3.6 Root

```
HitboxRoot Root
```

Reference to the top-level [HitboxRoot](#) component for this [NetworkObject](#).

6.35.3.7 SphereRadius

```
float SphereRadius
```

When [Type](#) is set to [HitboxTypes.Sphere](#), this defines the local-space geometry for narrow-phase checks.

6.35.3.8 Type

```
HitboxTypes Type
```

The collision geometry type for this [Hitbox](#).

6.35.4 Property Documentation

6.35.4.1 ColliderIndex

```
int ColliderIndex [get]
```

Index assigned to the collider of this hitbox on the lag-compensated snapshots.

6.35.4.2 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.

6.35.4.3 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](#).

6.35.4.4 Position

```
Vector3 Position [get]
```

World-space position (includes Offset) of this [Hitbox](#).

6.36 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.
- 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](#) 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.

- 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 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.

- 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.

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

6.36.1 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.

6.36.2 Member Function Documentation

6.36.2.1 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.

6.36.2.2 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.

6.36.2.3 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 <i>alpha</i> 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.

6.36.2.4 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.

6.36.2.5 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.

6.36.2.6 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> .
<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 <i>alpha</i> 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
```

6.36.2.7 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).
<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
```

6.36.2.8 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 <i>alpha</i> 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.

6.36.2.9 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.

6.36.2.10 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.

6.36.2.11 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 <i>alpha</i> value.

Parameters

<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

6.36.2.12 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). <small>Generated by Doxygen</small>

Returns

True if something is hit

6.36.2.13 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.

6.36.2.14 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
---------------	--------------------------------

Parameters

<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

6.36.2.15 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

6.36.3 Member Data Documentation

6.36.3.1 BVHDepth

`int BVHDepth`

Debug data from Broadphase BVH (tree depth).

6.36.3.2 BVHNodes

`int BVHNodes`

Debug data from Broadphase BVH (total nodes count).

6.36.3.3 DrawInfo

`LagCompensationDraw DrawInfo`

Debug data used to draw the BVH nodes and the lag compensation history.

6.36.3.4 TotalHitboxes

```
int TotalHitboxes
```

Debug data from lag compensation history (registered [Hitbox](#) count).

6.37 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 off 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 Spawner.
- 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

6.37.1 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.

6.37.2 Member Enumeration Documentation

6.37.2.1 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.
Generated by Doxygen	
Default	Ser of configuration flags with the default behaviour, suitable for most use-cases.

6.37.3 Member Function Documentation

6.37.3.1 DrawGizmos()

```
virtual void DrawGizmos (
    Color color,
    ref Matrix4x4 localToWorldMatrix ) [protected], [virtual]
```

Draws the gizmos for the [HitboxRoot](#)

6.37.3.2 InitHitboxes()

```
void InitHitboxes ( )
```

Finds child [Hitbox](#) components, and adds them to the [Hitboxes](#) collection.

6.37.3.3 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.

6.37.3.4 OnDrawGizmos()

```
void OnDrawGizmos ( )
```

[HitboxRoot](#) on draw gizmos.

6.37.3.5 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.

6.37.3.6 SetMinBoundingRadius()

```
void SetMinBoundingRadius ( )
```

Sets [BroadRadius](#) to a rough value which encompasses all [Hitboxes](#) in their current positions.

6.37.4 Member Data Documentation

6.37.4.1 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 off 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.

6.37.4.2 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.

6.37.4.3 GizmosColor

```
Color GizmosColor = Color.gray
```

Color used when drawing gizmos for this hitbox.

6.37.4.4 Hitboxes

```
Hitbox [] Hitboxes
```

All [Hitbox](#) instances in hierarchy. Auto-filled at Spawned.

6.37.4.5 MAX_HITBOXES

```
const Int32 MAX_HITBOXES = (sizeof(UInt32) * 8) - 1 [static]
```

The max number of hitboxes allowed under the same root.

6.37.4.6 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.

6.37.5 Property Documentation

6.37.5.1 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.

6.37.5.2 InInterest

```
bool InInterest [get]
```

If this [HitboxRoot](#) is in interest for the local player.

6.37.5.3 Manager

```
HitboxManager Manager [get]
```

Reference to associated hitbox manager (from which lag compensated queries can be performed).

6.38 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

6.38.1 Detailed Description

Project configuration settings specific to how the Host Migration behaves.

6.38.2 Member Data Documentation

6.38.2.1 EnableAutoUpdate

```
bool EnableAutoUpdate
```

Enabled the Host Migration feature

6.38.2.2 UpdateDelay

```
int UpdateDelay = 10
```

Delay between Host Migration Snapshot updates

6.39 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

6.39.1 Detailed Description

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

6.39.2 Property Documentation

6.39.2.1 GameMode

```
GameMode GameMode [get]
```

New GameMode the local peer will assume after the Host Migration

6.40 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.

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.

6.40.1 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.

6.40.2 Member Function Documentation

6.40.2.1 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.

6.41 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.

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.

6.41.1 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.

6.41.2 Member Function Documentation

6.41.2.1 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.

6.42 IAfterHostMigration Interface Reference

Used to mark NetworkBehaviors that need to be react after a Host Migration process

Public Member Functions

- void [AfterHostMigration](#) ()

Invoked after the Host Migration happens in order to setup non-networked data on NetworkBehaviors

6.42.1 Detailed Description

Used to mark NetworkBehaviors that need to be react after a Host Migration process

6.42.2 Member Function Documentation

6.42.2.1 AfterHostMigration()

```
void AfterHostMigration ( )
```

Invoked after the Host Migration happens in order to setup non-networked data on NetworkBehaviors

6.43 IAfterRender Interface Reference

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

Public Member Functions

- void [AfterRender](#) ()

Called after the render loop.

6.43.1 Detailed Description

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

6.43.2 Member Function Documentation

6.43.2.1 AfterRender()

```
void AfterRender ( )
```

Called after the render loop.

6.44 IAfterSpawned Interface Reference

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

Public Member Functions

- void [AfterSpawned](#) ()
Called after the object is spawned.

6.44.1 Detailed Description

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

6.44.2 Member Function Documentation

6.44.2.1 AfterSpawned()

```
void AfterSpawned ( )
```

Called after the object is spawned.

6.45 IAfterTick Interface Reference

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

Inherited by [HitboxManager](#).

Public Member Functions

- void [AfterTick \(\)](#)
Called after each tick simulation completes.

6.45.1 Detailed Description

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

6.45.2 Member Function Documentation

6.45.2.1 AfterTick()

```
void AfterTick ( )
```

Called after each tick simulation completes.

6.46 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.

Public Member Functions

- void [AfterUpdate \(\)](#)
Called at the end of the [Fusion](#) Update loop, before all Unity MonoBehaviour.Update() callbacks.

6.46.1 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.

6.46.2 Member Function Documentation

6.46.2.1 AfterUpdate()

```
void AfterUpdate ( )
```

Called at the end of the [Fusion](#) Update loop, before all Unity MonoBehaviour.Update() callbacks.

6.47 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.

6.47.1 Detailed Description

Defines an asynchronous operation.

6.47.2 Property Documentation

6.47.2.1 Error

`ExceptionDispatchInfo Error [get]`

Gets the exception information if an error occurred during the operation.

6.47.2.2 IsDone

`bool IsDone [get]`

Gets a value indicating whether the operation is done.

6.47.3 Event Documentation

6.47.3.1 Completed

Action<[IAsyncOperation](#)> Completed

Occurs when the operation is completed.

6.48 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.

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.

6.48.1 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.

6.48.2 Member Function Documentation

6.48.2.1 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.

6.49 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.

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.

6.49.1 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.

6.49.2 Member Function Documentation

6.49.2.1 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.

6.50 IBeforeCopyPreviousState Interface Reference

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

Inherited by [NetworkTransform](#).

Public Member Functions

- void [BeforeCopyPreviousState \(\)](#)

Called before the copy of the previous state.

6.50.1 Detailed Description

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

6.50.2 Member Function Documentation

6.50.2.1 BeforeCopyPreviousState()

```
void BeforeCopyPreviousState ( )
```

Called before the copy of the previous state.

6.51 IBeforeHitboxRegistration Interface Reference

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

Public Member Functions

- void [BeforeHitboxRegistration](#) ()

Called immediately before the [HitboxManager](#) registers hitboxes in a snapshot.

6.51.1 Detailed Description

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

6.51.2 Member Function Documentation

6.51.2.1 BeforeHitboxRegistration()

```
void BeforeHitboxRegistration ( )
```

Called immediately before the [HitboxManager](#) registers hitboxes in a snapshot.

6.52 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.

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.

6.52.1 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.

6.52.2 Member Function Documentation

6.52.2.1 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

<i>forwardTickCount</i>	How many forward ticks are going to be processed.
-------------------------	---

6.53 IBeforeTick Interface Reference

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

Public Member Functions

- void [BeforeTick](#) ()

Called before each tick is simulated.

6.53.1 Detailed Description

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

6.53.2 Member Function Documentation

6.53.2.1 BeforeTick()

```
void BeforeTick ( )
```

Called before each tick is simulated.

6.54 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.

Public Member Functions

- void [BeforeUpdate](#) ()
Called at the start of the [Fusion](#) Update loop, before the [Fusion](#) simulation loop.

6.54.1 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.

6.54.2 Member Function Documentation

6.54.2.1 BeforeUpdate()

```
void BeforeUpdate ( )
```

Called at the start of the [Fusion](#) Update loop, before the [Fusion](#) simulation loop.

6.55 ICOROUTINE Interface Reference

Defines a coroutine.

Inherits [IAsyncOperation](#), and [IEnumerator](#).

Additional Inherited Members

6.55.1 Detailed Description

Defines a coroutine.

6.56 IDespawned Interface Reference

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

Inherited by [NetworkBehaviour](#).

Public Member Functions

- void [Despawned](#) ([NetworkRunner](#) runner, bool hasState)

Called when a NetworkBehaviour is despawned.

6.56.1 Detailed Description

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

6.56.2 Member Function Documentation

6.56.2.1 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](#).

6.57 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.

6.57.1 Detailed Description

Defines the interface for reading and writing elements in a byte array.

Template Parameters

<i>T</i>	The type of the elements.
----------	---------------------------

6.57.2 Member Function Documentation

6.57.2.1 GetElementHashCode()

```
int GetElementHashCode (
    T element )
```

Calculate the hash code of an element.

Parameters

<i>element</i>	
----------------	--

Returns

6.57.2.2 GetElementWordCount()

```
int GetElementWordCount ( )
```

Gets the word count of an element.

Returns

The word count of an element.

6.57.2.3 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.

6.57.2.4 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.

6.57.2.5 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.

6.58 IFixedStorage Interface Reference

Interface for fixed storage types.

Inherited by [_128](#), [_16](#), [_2](#), [_256](#), [_32](#), [_4](#), [_512](#), [_64](#), and [_8](#).

6.58.1 Detailed Description

Interface for fixed storage types.

6.59 IIInputAuthorityGained Interface Reference

Interface for handling the event when the input authority is gained.

Public Member Functions

- void [InputAuthorityGained](#) ()
Method to be called when the input authority is gained.

6.59.1 Detailed Description

Interface for handling the event when the input authority is gained.

6.59.2 Member Function Documentation

6.59.2.1 InputAuthorityGained()

```
void InputAuthorityGained ( )
```

Method to be called when the input authority is gained.

6.60 IInputAuthorityLost Interface Reference

Interface for handling the event when the input authority is lost.

Public Member Functions

- void [InputAuthorityLost \(\)](#)
Method to be called when the input authority is lost.

6.60.1 Detailed Description

Interface for handling the event when the input authority is lost.

6.60.2 Member Function Documentation

6.60.2.1 InputAuthorityLost()

```
void InputAuthorityLost ( )
```

Method to be called when the input authority is lost.

6.61 IInterestEnter Interface Reference

Interface for handling the event when a player enters the area of interest.

Public Member Functions

- void [InterestEnter \(PlayerRef player\)](#)
Method to be called when a player enters the area of interest.

6.61.1 Detailed Description

Interface for handling the event when a player enters the area of interest.

6.61.2 Member Function Documentation

6.61.2.1 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.
---------------	--

6.62 IIInterestExit Interface Reference

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

Public Member Functions

- void [InterestExit](#) ([PlayerRef](#) player)
Method to be called when a player exits the area of interest.

6.62.1 Detailed Description

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

6.62.2 Member Function Documentation

6.62.2.1 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.
---------------	---

6.63 ILocalPrefabCreated Interface Reference

Interface for handling the event when a local prefab is created.

Public Member Functions

- void [LocalPrefabCreated](#) ()
Method to be called when a local prefab is created.

6.63.1 Detailed Description

Interface for handling the event when a local prefab is created.

6.63.2 Member Function Documentation

6.63.2.1 LocalPrefabCreated()

```
void LocalPrefabCreated ( )
```

Method to be called when a local prefab is created.

6.64 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.

6.64.1 Detailed Description

Defines the interface for a networked array.

6.64.2 Property Documentation

6.64.2.1 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.
--------------	--

6.65 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.

6.65.1 Detailed Description

Interface for a network asset source.

Template Parameters

<i>T</i>	Type of the network asset.
----------	----------------------------

6.65.2 Member Function Documentation

6.65.2.1 Acquire()

```
void Acquire (
    bool synchronous )
```

Acquires the network asset.

Parameters

<code>synchronous</code>	If true, the acquisition is done synchronously.
--------------------------	---

6.65.2.2 Release()

```
void Release ( )
```

Releases the network asset.

6.65.2.3 WaitForResult()

```
T WaitForResult ( )
```

Waits for the result of the network asset acquisition.

Returns

The network asset.

6.65.3 Property Documentation**6.65.3.1 Description**

```
string Description [get]
```

Gets the description of the network asset.

6.65.3.2 IsCompleted

```
bool IsCompleted [get]
```

Checks if the network asset acquisition is completed.

6.66 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.

6.66.1 Detailed Description

Defines the interface for a networked dictionary.

6.66.2 Member Function Documentation

6.66.2.1 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 >](#).

6.67 INetworkInput Interface Reference

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

6.67.1 Detailed Description

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

6.68 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.

6.68.1 Detailed Description

Defines the interface for a networked linked list.

6.68.2 Member Function Documentation

6.68.2.1 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 >](#).

6.69 INetworkObjectInitializer Interface Reference

Interface for initializing network objects.

Inherited by [NetworkObjectInitializerUnity](#).

Public Member Functions

- void [InitializeNetworkState](#) ([NetworkObject](#) networkObject)
Initializes the network object.

6.69.1 Detailed Description

Interface for initializing network objects.

6.69.2 Member Function Documentation

6.69.2.1 InitializeNetworkState()

```
void InitializeNetworkState (
    NetworkObject networkObject )
```

Initializes the network object.

Parameters

<code>networkObject</code>	The network object to initialize.
----------------------------	-----------------------------------

Implemented in [NetworkObjectInitializerUnity](#).

6.70 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.
- `void ReleaseInstance (NetworkRunner runner, in NetworkObjectReleaseContext context)`
Releases an instance of a network object.

6.70.1 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()`.

6.70.2 Member Function Documentation

6.70.2.1 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](#).

6.70.2.2 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](#).

6.71 INetworkPrefabSource Interface Reference

Interface for a network prefab source.

Inherits [INetworkAssetSource< NetworkObject >](#).

Properties

- [NetworkObjectGuid AssetGuid](#) [get]
Gets the G UID of the network object asset.

Additional Inherited Members**6.71.1 Detailed Description**

Interface for a network prefab source.

6.71.2 Property Documentation

6.71.2.1 AssetGuid

```
NetworkObjectGuid AssetGuid [get]
```

Gets the GUID of the network object asset.

6.72 INetworkRunnerCallbacks Interface Reference

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

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

6.72.1 Detailed Description

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

6.72.2 Member Function Documentation

6.72.2.1 OnConnectedToServer()

```
void OnConnectedToServer (
    NetworkRunner runner )
```

Callback when `NetworkRunner` successfully connects to a server or host.

6.72.2.2 OnConnectFailed()

```
void OnConnectFailed (
    NetworkRunner runner,
    NetAddress remoteAddress,
    NetConnectFailedReason reason )
```

Callback when `NetworkRunner` fails to connect to a server or host.

6.72.2.3 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

6.72.2.4 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

6.72.2.5 OnDisconnectedFromServer()

```
void OnDisconnectedFromServer (
    NetworkRunner runner,
    NetDisconnectReason reason )
```

Callback when [NetworkRunner](#) disconnects from a server or host.

6.72.2.6 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

6.72.2.7 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 */ });
```

6.72.2.8 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

6.72.2.9 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

6.72.2.10 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

6.72.2.11 OnPlayerJoined()

```
void OnPlayerJoined (
    NetworkRunner runner,
    PlayerRef player )
```

Callback from a [NetworkRunner](#) when a new player has joined.

6.72.2.12 OnPlayerLeft()

```
void OnPlayerLeft (
    NetworkRunner runner,
    PlayerRef player )
```

Callback from a [NetworkRunner](#) when a player has disconnected.

6.72.2.13 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

6.72.2.14 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

6.72.2.15 OnSceneLoadDone()

```
void OnSceneLoadDone (
    NetworkRunner runner )
```

Callback is invoked when a Scene Load has finished

Parameters

<i>runner</i>	NetworkRunner reference
---------------	-------------------------

6.72.2.16 OnSceneLoadStart()

```
void OnSceneLoadStart (
    NetworkRunner runner )
```

Callback is invoked when a Scene Load has started

Parameters

<i>runner</i>	NetworkRunner reference
---------------	-------------------------

6.72.2.17 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

6.72.2.18 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

6.72.2.19 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

6.73 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.

6.73.1 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 LateUpdate.

6.73.2 Member Function Documentation

6.73.2.1 Initialize()

```
void Initialize (
    NetworkRunner runner )
```

Called when the [NetworkRunner](#) is started.

Parameters

runner	The NetworkRunner instance.
------------------------	---

6.73.2.2 Shutdown()

```
void Shutdown (
    NetworkRunner runner )
```

Called when the [NetworkRunner](#) is stopped.

Parameters

runner	The NetworkRunner instance.
------------------------	---

6.74 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

6.74.1 Detailed Description

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

6.74.2 Member Function Documentation

6.74.2.1 GetSceneRef() [1/2]

```
SceneRef GetSceneRef (
    GameObject gameObject )
```

Gets a [SceneRef](#) for the scene that the given GameObject belongs to.

6.74.2.2 GetSceneRef() [2/2]

```
SceneRef GetSceneRef (
    string sceneNameOrPath )
```

Gets a [SceneRef](#) for the given scene name or path.

6.74.2.3 Initialize()

```
void Initialize (
    NetworkRunner runner )
```

Callback for initialization

6.74.2.4 IsRunnerScene()

```
bool IsRunnerScene (
    Scene scene )
```

Signals if the given scene is the main runner scene. Mostly used for Multipeer logic

6.74.2.5 LoadScene()

```
NetworkSceneAsyncOp LoadScene (
    SceneRef sceneRef,
    NetworkLoadSceneParameters parameters )
```

Loads a given scene with the specified parameters.

Returns

Returns a [NetworkSceneAsyncOp](#) that can be waited

6.74.2.6 MakeDontDestroyOnLoad()

```
void MakeDontDestroyOnLoad (
    GameObject obj )
```

Mark an object as DontDestroyOnLoad.

6.74.2.7 MoveGameObjectToScene()

```
bool MoveGameObjectToScene (
    GameObject gameObject,
    SceneRef sceneRef )
```

Move a GameObject to a desired scene.

Returns

Return true if the operation was successfully

6.74.2.8 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

6.74.2.9 Shutdown()

```
void Shutdown ( )
```

Callback for shutdown and clean up

6.74.2.10 TryGetPhysicsScene2D()

```
bool TryGetPhysicsScene2D (
    out PhysicsScene2D scene2D )
```

Tries to get the physics scene 2D.

Returns

Returns true if the operation was successfully

6.74.2.11 TryGetPhysicsScene3D()

```
bool TryGetPhysicsScene3D (
    out PhysicsScene scene3D )
```

Tries to get the physics scene 3D.

Returns

Returns true if the operation was successfully

6.74.2.12 UnloadScene()

```
NetworkSceneAsyncOp UnloadScene (
    SceneRef sceneRef )
```

Unloads a given scene.

Returns

Returns a [NetworkSceneAsyncOp](#) that can be waited

6.74.3 Property Documentation

6.74.3.1 IsBusy

```
bool IsBusy [get]
```

Signals if the [INetworkSceneManager](#) instance is busy with any scene loading operations

6.74.3.2 MainRunnerScene

```
Scene MainRunnerScene [get]
```

The main scene of the [NetworkRunner](#). Mostly used for Multipeer logic

6.75 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](#).

6.75.1 Detailed Description

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

6.76 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.

6.76.1 Detailed Description

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

6.76.2 Member Function Documentation

6.76.2.1 Teleport()

```
void Teleport (
    Vector3? position = null,
    Quaternion? rotation = null )
```

Teleports to the indicated values, and network the Teleport event.

Implemented in [NetworkTransform](#).

6.77 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.

6.77.1 Detailed Description

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

6.77.2 Member Function Documentation

6.77.2.1 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 >](#)

6.77.2.2 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 >](#)

6.78 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

6.78.1 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.
----------	-----------------------

6.78.2 Property Documentation

6.78.2.1 DataProperty

`T DataProperty [get], [set]`

Gets or sets the data for the Unity value surrogate.

6.79 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`.

6.79.1 Detailed Description

A base class for Unity array surrogates.

Template Parameters

<code>T</code>	Unmanaged type of the array elements.
<code>ReaderWriter</code>	Unmanaged type of the reader/writer for the array elements.

Type Constraints

`T : unmanaged`

`ReaderWriter : unmanaged`

`ReaderWriter : IElementReaderWriter<T>`

6.79.2 Member Function Documentation

6.79.2.1 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](#).

6.79.2.2 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](#).

6.79.2.3 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](#).

6.79.3 Property Documentation

6.79.3.1 DataProperty

```
abstract T [] DataProperty [get], [set]
```

Gets or sets the data array for the [UnityArraySurrogate](#).

6.80 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.

6.80.1 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*>*

6.80.2 Member Function Documentation

6.80.2.1 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](#).

6.80.2.2 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](#).

6.80.2.3 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](#).

6.80.3 Property Documentation

6.80.3.1 DataProperty

```
abstract SerializableDictionary<TKeyType, TValueType> DataProperty [get], [set]
```

Gets or sets the data property.

6.81 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.

6.81.1 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*>**6.81.2 Member Function Documentation****6.81.2.1 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](#).

6.81.2.2 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](#).

6.81.2.3 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](#).

6.81.3 Property Documentation

6.81.3.1 DataProperty

```
abstract T [] DataProperty [get], [set]
```

Gets or sets the data property.

6.82 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.

6.82.1 Detailed Description

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

6.82.2 Member Function Documentation

6.82.2.1 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 >](#).

6.82.2.2 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 >](#).

6.82.2.3 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 >](#).

6.83 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](#).

6.83.1 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*>

6.83.2 Member Function Documentation

6.83.2.1 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](#).

6.83.2.2 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](#).

6.83.2.3 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](#).

6.83.3 Property Documentation

6.83.3.1 DataProperty

```
abstract T DataProperty [get], [set]
```

Gets or sets the data for the [UnityValueSurrogate](#).

6.84 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

6.84.1 Detailed Description

A set of parameters that tune the interpolated correction of prediction error on transform data.

6.84.2 Member Data Documentation

6.84.2.1 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.

6.84.2.2 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.

6.84.2.3 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.

6.84.2.4 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.

6.84.2.5 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.

6.84.2.6 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.

6.84.2.7 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](#).

6.84.2.8 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](#).

6.84.2.9 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.

6.85 IPlayerJoined Interface Reference

Interface for handling the event when a player joins the game.

Public Member Functions

- void [PlayerJoined](#) ([PlayerRef](#) player)
Method to be called when a player joins the game.

6.85.1 Detailed Description

Interface for handling the event when a player joins the game.

6.85.2 Member Function Documentation

6.85.2.1 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.
---------------	---------------------------------

6.86 IPlayerLeft Interface Reference

Interface for handling the event when a player leaves the game.

Public Member Functions

- void [PlayerLeft](#) ([PlayerRef](#) player)
Method to be called when a player leaves the game.

6.86.1 Detailed Description

Interface for handling the event when a player leaves the game.

6.86.2 Member Function Documentation

6.86.2.1 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.
---------------	-------------------------------

6.87 IRemotePrefabCreated Interface Reference

Interface for handling the event when a remote prefab is created.

Public Member Functions

- void [RemotePrefabCreated](#) ()
Method to be called when a remote prefab is created.

6.87.1 Detailed Description

Interface for handling the event when a remote prefab is created.

6.87.2 Member Function Documentation

6.87.2.1 RemotePrefabCreated()

```
void RemotePrefabCreated ( )
```

Method to be called when a remote prefab is created.

6.88 ISceneLoadDone Interface Reference

Interface for handling the event when a scene load operation is completed.

Public Member Functions

- void [SceneLoadDone](#) (in [SceneLoadDoneArgs](#) sceneInfo)
Method to be called when a scene load operation is completed.

6.88.1 Detailed Description

Interface for handling the event when a scene load operation is completed.

6.88.2 Member Function Documentation

6.88.2.1 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.
------------------------	---

6.89 ISceneLoadStart Interface Reference

Interface for handling the event when a scene load operation is started.

Public Member Functions

- void [SceneLoadStart](#) ([SceneRef](#) sceneRef)
Method to be called when a scene load operation is started.

6.89.1 Detailed Description

Interface for handling the event when a scene load operation is started.

6.89.2 Member Function Documentation

6.89.2.1 SceneLoadStart()

```
void SceneLoadStart (
    SceneRef sceneRef )
```

Method to be called when a scene load operation is started.

Parameters

sceneRef	Reference to the scene that is being loaded.
----------	--

6.90 ISimulationEnter Interface Reference

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

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.

6.90.1 Detailed Description

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

6.90.2 Member Function Documentation

6.90.2.1 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.

6.91 ISimulationExit Interface Reference

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

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.

6.91.1 Detailed Description

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

6.91.2 Member Function Documentation

6.91.2.1 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.

6.92 ISpawned Interface Reference

Interface for handling the event when an object is spawned.

Inherited by [HitboxManager](#), and [NetworkBehaviour](#).

Public Member Functions

- void [Spawned](#) ()

Method to be called when an object is spawned.

6.92.1 Detailed Description

Interface for handling the event when an object is spawned.

6.92.2 Member Function Documentation

6.92.2.1 Spawned()

```
void Spawned ( )
```

Method to be called when an object is spawned.

Implemented in [NetworkTransform](#), [NetworkMecanimAnimator](#), and [NetworkBehaviour](#).

6.93 IStateAuthorityChanged Interface Reference

Interface for handling the event when the state authority changes.

Public Member Functions

- void [StateAuthorityChanged \(\)](#)
Method to be called when the state authority changes.

6.93.1 Detailed Description

Interface for handling the event when the state authority changes.

6.93.2 Member Function Documentation

6.93.2.1 StateAuthorityChanged()

```
void StateAuthorityChanged ( )
```

Method to be called when the state authority changes.

6.94 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).

6.94.1 Detailed Description

Defines a lag compensated query hit result.

6.94.2 Member Function Documentation

6.94.2.1 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.

6.94.2.2 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.

6.94.3 Member Data Documentation**6.94.3.1 Collider**

Collider Collider

PhysX collider hit. Null in case hit is a [Fusion Hitbox](#) or a Box2D hit.

6.94.3.2 Collider2D

Collider2D Collider2D

Box2D collider hit. Null in case hit is a [Fusion Hitbox](#) or a PhysX hit.

6.94.3.3 Distance

```
float Distance
```

Distance (if requested) to hit, at the lag compensated time.

6.94.3.4 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.

6.94.3.5 Hitbox

```
Hitbox Hitbox
```

Fusion's [Hitbox](#). Null in case the hit was on PhysX or Box2D.

6.94.3.6 HitboxColliderPosition

```
Vector3 HitboxColliderPosition
```

The hitbox collider position on the snapshot.

6.94.3.7 HitboxColliderRotation

```
Quaternion HitboxColliderRotation
```

The hitbox collider rotation on the snapshot.

6.94.3.8 Normal

```
Vector3 Normal
```

Surface normal (if requested) of the hit, at the lag compensated time.

6.94.3.9 Point

`Vector3 Point`

Point of impact of the hit, at the lag compensated time.

6.94.3.10 Type

`HitType Type`

Hit object source (PhysX or Fusion Hitboxes).

6.95 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.

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](#).

6.95.1 Detailed Description

Represents an Axis-Aligned Bounding Box ([AABB](#)).

6.95.2 Constructor & Destructor Documentation

6.95.2.1 [AABB\(\)](#) [1/2]

```
AABB (
    Bounds bounds )
```

Constructs an [AABB](#) from a Unity `Bounds` object.

Parameters

<i>bounds</i>	The Unity Bounds object to construct the AABB from.
---------------	---

6.95.2.2 [AABB\(\)](#) [2/2]

```
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 .

6.95.3 Member Data Documentation**6.95.3.1 Center**

```
readonly Vector3 Center
```

Represents the center of the [AABB](#).

6.95.3.2 Extents

```
readonly Vector3 Extents
```

Represents the extents (half-widths) of the [AABB](#).

6.95.3.3 Max

```
readonly Vector3 Max
```

Represents the maximum point (upper corner) of the [AABB](#).

6.95.3.4 Min

readonly Vector3 Min

Represents the minimum point (lower corner) of the [AABB](#).

6.96 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.

6.96.1 Detailed Description

Class that represents a box overlap query. Used to query against the [NetworkRunner.LagCompensation API](#).

6.96.2 Constructor & Destructor Documentation

6.96.2.1 [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.
-------------------------	--

6.96.2.2 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.

6.96.3 Member Function Documentation**6.96.3.1 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](#).

6.96.4 Member Data Documentation

6.96.4.1 Center

Vector3 Center

The box query center.

6.96.4.2 Extents

Vector3 Extents

The box query extents.

6.96.4.3 Rotation

Quaternion Rotation

The box query rotation.

6.97 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.

6.97.1 Detailed Description

Base parameters needed to execute a box overlap query

6.97.2 Constructor & Destructor Documentation

6.97.2.1 BoxOverlapQueryParams()

```
BoxOverlapQueryParams (
    QueryParams queryParams,
    Vector3 center,
    Vector3 extents,
    Quaternion rotation,
    int staticHitsCapacity )
```

Create a new `BoxOverlapQueryParams`

Parameters

<code>queryParams</code>	Parameters to be used
<code>center</code>	The query center
<code>extents</code>	The query extents
<code>rotation</code>	The query rotation
<code>staticHitsCapacity</code>	Capacity for the cached PhysX and Box2D static hits.

6.97.3 Member Data Documentation

6.97.3.1 Center

`Vector3 Center`

Represents the center of the box for the overlap query.

6.97.3.2 Extents

`Vector3 Extents`

Represents the extents of the box for the overlap query.

6.97.3.3 QueryParams

`QueryParams` `QueryParams`

Represents the base parameters for the query.

6.97.3.4 Rotation

`Quaternion` `Rotation`

Represents the rotation of the box for the overlap query.

6.97.3.5 StaticHitsCapacity

`int` `StaticHitsCapacity`

Represents the capacity for the cached PhysX and Box2D static hits.

6.98 BVHDraw Class Reference

Provide a way to iterate over BVH and return a [BVHNodeDrawInfo](#) for each node.

Inherits `IEnumerable< BVHNodeDrawInfo >`.

Public Member Functions

- `IEnumerator< BVHNodeDrawInfo > GetEnumerator ()`
Returns an enumerator that iterates through the [BVHNodeDrawInfo](#).

6.98.1 Detailed Description

Provide a way to iterate over BVH and return a [BVHNodeDrawInfo](#) for each node.

6.98.2 Member Function Documentation

6.98.2.1 GetEnumerator()

`IEnumerator<BVHNodeDrawInfo>` `GetEnumerator ()`

Returns an enumerator that iterates through the [BVHNodeDrawInfo](#).

6.99 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

6.99.1 Detailed Description

Container class to provide the necessary info to draw nodes from the BVH

6.99.2 Property Documentation

6.99.2.1 Bounds

`Bounds Bounds [get]`

Get the node Bounds

6.99.2.2 Depth

`int Depth [get]`

Get the node depth on the BVH

6.99.2.3 MaxDepth

`int MaxDepth [get]`

Get the BVH max depth

6.100 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.

6.100.1 Detailed Description

Container class to provide the necessary information to draw a hitbox collider

6.100.2 Property Documentation

6.100.2.1 BoxExtents

`Vector3 BoxExtents [get]`

The box extends of the collider Used on [HitboxTypes](#) of types: Box

6.100.2.2 CapsuleBottomCenter

`Vector3 CapsuleBottomCenter [get]`

Represents the bottom center position of the capsule collider.

6.100.2.3 CapsuleExtents

```
float CapsuleExtents [get]
```

The height for capsule colliders.

See also

[HitboxTypes](#)

6.100.2.4 CapsuleTopCenter

```
Vector3 CapsuleTopCenter [get]
```

Represents the top center position of the capsule collider.

6.100.2.5 LocalToWorldMatrix

```
Matrix4x4 LocalToWorldMatrix [get]
```

The local to world matrix of the collider.

6.100.2.6 Offset

```
Vector3 Offset [get]
```

The offset of the collider.

6.100.2.7 Radius

```
float Radius [get]
```

The radius of the collider. Used on [HitboxTypes](#) of types: Sphere and Capsule.

6.100.2.8 Type

```
HitboxTypes Type [get]
```

The [HitboxTypes](#) of the collider.

6.101 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.

6.101.1 Detailed Description

Provide a way to iterate over the HitboxBuffer.HitboxSnapshot and return the [ColliderDrawInfo](#) for each collider on the snapshot.

6.101.2 Member Function Documentation

6.101.2.1 GetEnumerator()

[IEnumerator<ColliderDrawInfo> GetEnumerator \(\)](#)

Returns an enumerator that iterates through the [ColliderDrawInfo](#) of this container.

6.102 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.

6.102.1 Detailed Description

LagCompensated Extension methods

6.102.2 Member Function Documentation

6.102.2.1 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.
-------------------------------	--------------------------

6.102.2.2 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.
-------------------------------	--------------------------

6.103 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.

6.103.1 Detailed Description

Provide access to iterate over the lag compensation system components and give the necessary information to draw them.

6.103.2 Member Function Documentation

6.103.2.1 `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

6.103.3 Member Data Documentation

6.103.3.1 `BVHDraw`

`BVHDraw BVHDraw`

Iterate over to get the BVH node draw data.

6.103.3.2 `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.

6.104 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.

6.104.1 Detailed Description

Details regarding a shape intersection. It does not carry information about the intersection happening or not.

6.104.2 Member Data Documentation

6.104.2.1 Normal

`Vector3 Normal`

Vector that described the plane of smallest penetration between the shapes.

6.104.2.2 Penetration

`float Penetration`

Penetration along the normal plane.

6.104.2.3 Point

`Vector3 Point`

Contact point.

6.105 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.

6.105.1 Detailed Description

[Query](#) parameters for position rotation query

6.105.2 Constructor & Destructor Documentation

6.105.2.1 PositionRotationQueryParams()

```
PositionRotationQueryParams (
    queryParams queryParams,
    Hitbox hitbox )
```

Create a new PositionRotationQueryParams.

Parameters

<code>queryParams</code>	Parameters to be used
<code>hitbox</code>	The hitbox to be queried

6.105.3 Member Data Documentation

6.105.3.1 Hitbox

[Hitbox Hitbox](#)

Represents the hitbox to be queried.

6.105.3.2 QueryParams

`QueryParams` `QueryParams`

Represents the base parameters for the query.

6.106 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? `TickCount`
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`.

6.106.1 Detailed Description

Base class for all Lag Compensation queries

6.106.2 Constructor & Destructor Documentation

6.106.2.1 Query()

```
Query (
    ref QueryParams qParams ) [protected]
```

Initializes a new instance of the [Query](#) class using the provided [QueryParams](#).

Parameters

<i>queryParams</i>	The QueryParams to use for initializing the Query .
--------------------	---

6.106.3 Member Function Documentation

6.106.3.1 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](#).

6.106.4 Member Data Documentation

6.106.4.1 Alpha

```
float? Alpha
```

Represents the interpolation factor between the current and next simulation tick.

6.106.4.2 LayerMask

```
LayerMask LayerMask
```

Represents the layer mask to selectively ignore colliders when performing the query.

6.106.4.3 Options

`HitOptions` Options

Represents the options for the hit detection of the query.

6.106.4.4 Player

`PlayerRef` Player

Represents the player who initiated the query.

6.106.4.5 PreProcessingDelegate

`PreProcessingDelegate` PreProcessingDelegate

Represents the delegate to be called for pre-processing before the query is performed.

6.106.4.6 Tick

`int?` `Tick`

Represents the simulation tick at which the query was initiated.

6.106.4.7 TickTo

`int?` `TickTo`

Represents the simulation tick to which the query is performed.

6.106.4.8 TriggerInteraction

`QueryTriggerInteraction` TriggerInteraction

Represents the interaction type of the query with triggers.

6.106.4.9 UserArgs

```
void* UserArgs
```

Represents the user arguments for the query.

6.107 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.

6.107.1 Detailed Description

Base parameters needed to execute a query.

6.107.2 Member Data Documentation

6.107.2.1 Alpha

```
float? Alpha
```

Represents the interpolation factor between the current and next simulation tick.

6.107.2.2 LayerMask

`LayerMask` `LayerMask`

Represents the layer mask to selectively ignore colliders when performing the query.

6.107.2.3 Options

`HitOptions` `Options`

Represents the options for the hit detection of the query.

6.107.2.4 Player

`PlayerRef` `Player`

Represents the player who initiated the query.

6.107.2.5 PreProcessingDelegate

`PreProcessingDelegate` `PreProcessingDelegate`

Represents the delegate to be called for pre-processing before the query is performed.

6.107.2.6 Tick

`int` `Tick`

Represents the simulation tick at which the query was initiated.

6.107.2.7 TickTo

`int?` `TickTo`

Represents the simulation tick to which the query is performed.

6.107.2.8 TriggerInteraction

```
QueryTriggerInteraction TriggerInteraction
```

Represents the interaction type of the query with triggers.

6.107.2.9 UserArgs

```
void* UserArgs
```

Represents the user arguments for the query.

6.108 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

6.108.1 Detailed Description

Class that represents a raycast all query. Used to query against the [NetworkRunner.LagCompensation](#) API.

6.108.2 Constructor & Destructor Documentation

6.108.2.1 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.
---------------------------------	--

6.108.2.2 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.

6.109 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.

6.109.1 Detailed Description

Class that represents a raycast query. Used to query against the [NetworkRunner.LagCompensation](#) API.

6.109.2 Constructor & Destructor Documentation

6.109.2.1 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.
---------------------------	--

6.109.3 Member Function Documentation

6.109.3.1 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](#).

6.109.4 Member Data Documentation

6.109.4.1 Direction

Vector3 **Direction**

Represents the direction of the raycast for the query.

6.109.4.2 Length

float **Length**

Represents the maximum length of the raycast for the query.

6.109.4.3 Origin

Vector3 **Origin**

Represents the origin point of the raycast for the query.

6.110 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.

6.110.1 Detailed Description

Base parameters needed to execute a raycast query

6.110.2 Constructor & Destructor Documentation

6.110.2.1 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.

6.110.3 Member Data Documentation

6.110.3.1 Direction

```
Vector3 Direction
```

Represents the direction of the raycast for the query.

6.110.3.2 Length

```
float Length
```

Represents the maximum length of the raycast for the query.

6.110.3.3 Origin

Vector3 Origin

Represents the origin point of the raycast for the query.

6.110.3.4 QueryParams

QueryParams QueryParams

Represents the base parameters for the raycast query.

6.110.3.5 StaticHitsCapacity

int StaticHitsCapacity

Represents the capacity for the cached PhysX and Box2D static hits.

6.111 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](#).

6.111.1 Detailed Description

Provide a way to iterate over the HitboxBuffer and return the [HitboxColliderContainerDraw](#) container for each snapshot on the buffer.

6.111.2 Member Function Documentation

6.111.2.1 GetEnumerator()

```
IEnumerator<HitboxColliderContainerDraw> GetEnumerator ( )
```

Returns an enumerator that iterates through the [HitboxColliderContainerDraw](#).

6.112 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.

6.112.1 Detailed Description

Class that represents a sphere overlap query. Used to query against the [NetworkRunner.LagCompensation](#) API.

6.112.2 Constructor & Destructor Documentation

6.112.2.1 SphereOverlapQuery() [1/2]

```
SphereOverlapQuery (
    ref SphereOverlapQueryParams sphereOverlapParams )
```

Create a new [SphereOverlapQuery](#) with the given [SphereOverlapQueryParams](#).

Parameters

<i>sphereOverlapParams</i>	The parameters to be used when creating the query.
----------------------------	--

6.112.2.2 SphereOverlapQuery() [2/2]

```
SphereOverlapQuery (
    ref SphereOverlapQueryParams sphereOverlapParams,
    Collider[ ] physXOverlapHitsCache,
    Collider2D[ ] box2DOverlapHitsCache )
```

Create a new [SphereOverlapQuery](#) with the given [SphereOverlapQueryParams](#).

Parameters

<i>sphereOverlapParams</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.

6.112.3 Member Function Documentation**6.112.3.1 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](#).

6.112.4 Member Data Documentation

6.112.4.1 Center

Vector3 Center

Represents the center of the sphere for the overlap query.

6.112.4.2 Radius

float Radius

Represents the radius of the sphere for the overlap query.

6.113 SphereOverlapQueryParams Struct Reference

Base parameters needed to execute a sphere overlap query

Public Member Functions

- [SphereOverlapQueryParams \(QueryParams queryParams, Vector3 center, float radius, int staticHitsCapacity\)](#)
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.

6.113.1 Detailed Description

Base parameters needed to execute a sphere overlap query

6.113.2 Constructor & Destructor Documentation

6.113.2.1 [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.

6.113.3 Member Data Documentation

6.113.3.1 Center

`Vector3 Center`

Represents the center of the sphere for the overlap query.

6.113.3.2 QueryParams

`QueryParams queryParams`

Represents the base parameters for the sphere overlap query.

6.113.3.3 Radius

`float Radius`

Represents the radius of the sphere for the overlap query.

6.113.3.4 StaticHitsCapacity

`int StaticHitsCapacity`

Represents the capacity for the cached PhysX and Box2D static hits.

6.114 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.

6.114.1 Detailed Description

Settings for lag compensation history.

6.114.2 Member Data Documentation

6.114.2.1 CachedStaticCollidersSize

```
int CachedStaticCollidersSize = 64
```

The size of the cached static colliders (PhysX or Box2D) array of the default Lag Compensation Queries.

6.114.2.2 Enabled

```
bool Enabled = false
```

Indicates if a `HitboxManager` instance should be added when the `NetworkRunner` is initialized.

6.114.2.3 HitboxBufferLengthInMs

```
int HitboxBufferLengthInMs = 200
```

Hitbox snapshot history length in milliseconds.

6.114.2.4 HitboxDefaultCapacity

```
int HitboxDefaultCapacity = 512
```

`Hitbox` capacity per snapshot.

6.114.3 Property Documentation

6.114.3.1 ExpansionFactor

```
float ExpansionFactor [get]
```

Broadphase BVH node expansion factor (default 20%) for leaf nodes, so updates are not too frequent.

6.114.3.2 Optimize

```
bool Optimize [get]
```

Optional: tries to optimize broadphase BVH every update. May be removed in the future.

6.115 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

6.115.1 Detailed Description

Holds information about a Lobby

6.115.2 Property Documentation

6.115.2.1 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.

6.115.2.2 Name

```
string Name [get]
```

Lobby Name

6.115.2.3 Region

```
string Region [get]
```

Stores the current connected Region

6.116 LogSimpleUnity Class Reference

Log Simple Unity

Inherits [ILogger](#).

Public Member Functions

- void [Log](#) (LogType logType, object message, in LogContext logContext)
- void [LogException](#) (Exception ex, in LogContext logContext)

6.116.1 Detailed Description

Log Simple Unity

6.117 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 GetComponentsInChildren, 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 GetComponentsInChildren, 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 GetComponentsInChildren, 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.

6.117.1 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.

6.117.2 Member Function Documentation

6.117.2.1 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**

6.117.2.2 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**

6.117.2.3 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*6.117.2.4 **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*6.117.2.5 **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

6.117.2.6 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

6.117.2.7 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

6.117.2.8 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

T	The type of component to find.
TStopOn	The type of component that stops the search.

Parameters

t	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

6.117.2.9 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

6.117.2.10 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

T	Cast found components to this type. Typically Component, but any other class/interface will work as long as they are assignable from SearchT.
TSearch	Find components of this class or interface type.
TStop	When this component is found, no further recursing will be performed on that node.

Type Constraints

T : class

TSearch : class

6.117.2.11 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

6.117.2.12 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

6.117.2.13 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

6.117.2.14 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

6.118 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.
- [IEnumerator 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 [ToString \(\)](#)
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]

6.118.1 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

T	T can be a primitive, or an INetworkStruct .
----------	--

6.118.2 Constructor & Destructor Documentation

6.118.2.1 NetworkArray()

```
NetworkArray (
    byte * array,
    int length,
    IElementReaderWriter< T > readerWriter )
```

NetworkArray constructor.

6.118.3 Member Function Documentation

6.118.3.1 Clear()

```
void Clear( )
```

Clears the array, setting all values to default.

6.118.3.2 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 sourceOffset and sourceCount is greater than the length of the source list.

6.118.3.3 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.

6.118.3.4 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.

6.118.3.5 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> .
--------------------------	---

6.118.3.6 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.

6.118.3.7 Get()

```
T Get (
    int index )
```

Returns the array value at supplied index.

6.118.3.8 GetEnumerator()

```
Enumerator GetEnumerator ( )
```

Returns an enumerator that iterates through the collection.

6.118.3.9 operator NetworkArrayReadOnly< T >()

```
static implicit operator NetworkArrayReadOnly< T > (
    NetworkArray< T > value ) [static]
```

Returns a [NetworkArrayReadOnly](#) view of this array.

Parameters

value	NetworkArray to convert.
-------	--

Returns

[NetworkArrayReadOnly](#) view of this array.

6.118.3.10 Set()

```
T Set (
    int index,
    T value )
```

Sets the array value at the supplied index.

6.118.3.11 ToArray()

```
T [ ] ToArray ( )
```

Allocates a new array and copies values from this array. For a non-alloc alternative use [CopyTo\(List<T>\)](#).

6.118.3.12 ToStringString()

```
string ToStringString ( )
```

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.

6.118.3.13 ToReadOnly()

```
NetworkArrayReadOnly<T> ToReadOnly ( )
```

Returns a [NetworkArrayReadOnly](#) view of this array.

6.118.3.14 ToString()

```
override string ToString ( )
```

Returns a string that represents the current object.

6.118.4 Property Documentation

6.118.4.1 Length

```
int Length [get]
```

The fixed size of the array.

6.118.4.2 this[int index]

```
T this[int index] [get], [set]
```

Indexer of array elements.

6.119 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.

6.119.1 Detailed Description

Enumerator for [NetworkArray](#).

6.119.2 Constructor & Destructor Documentation

6.119.2.1 Enumerator()

```
Enumerator (
    NetworkArray< T > array )
```

Initializes a new instance of the [Enumerator](#) with the specified [NetworkArray](#).

Parameters

<code>array</code>	The NetworkArray to enumerate.
--------------------	--

6.119.3 Member Function Documentation

6.119.3.1 Dispose()

```
void Dispose ( )
```

Releases all resources used by the [Enumerator](#).

6.119.3.2 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.

6.119.3.3 Reset()

```
void Reset ( )
```

Sets the enumerator to its initial position, which is before the first element in the collection.

6.119.4 Property Documentation

6.119.4.1 Current [1/2]

```
T Current [get]
```

Gets the current element in the collection.

6.119.4.2 Current [2/2]

```
object IEnumarator. Current [get]
```

Gets the current element in the collection.

6.120 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](#).

6.120.1 Detailed Description

Provides extension methods for the [NetworkArray](#) class.

6.120.2 Member Function Documentation

6.120.2.1 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

6.120.2.2 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 >*

6.121 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.

6.121.1 Detailed Description

Provides a read-only view of a network array.

Template Parameters

<i>T</i>	The type of the elements.
----------	---------------------------

6.121.2 Property Documentation

6.121.2.1 Length

int Length [get]

The fixed size of the array.

6.121.2.2 this[int index]

T this[int index] [get]

Indexer of array elements.

6.122 NetworkAssemblyIgnoreAttribute Class Reference

Network Assembly Ignore Attribute

Inherits Attribute.

6.122.1 Detailed Description

Network Assembly Ignore Attribute

6.123 NetworkAssemblyWeavedAttribute Class Reference

Network Assembly Weaved Attribute

Inherits Attribute.

6.123.1 Detailed Description

Network Assembly Weaved Attribute

6.124 NetworkBehaviour Class Reference

Base class for [Fusion](#) network components, which are associated with a [NetworkObject](#).

Inherits [SimulationBehaviour](#), [ISpawned](#), and [IDespawned](#).

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 > ()
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](#) ([NetworkBehaviour](#) obj, byte *data)
Serializes a [NetworkBehaviour](#) into 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](#) ([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.

6.124.1 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.

6.124.2 Member Function Documentation

6.124.2.1 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.
------------------	--

6.124.2.2 CopyStateFrom()

```
void CopyStateFrom (
    NetworkBehaviour source )
```

Copies entire state of passed in source [NetworkBehaviour](#)

Parameters

<i>source</i>	Source NetworkBehaviour to copy data from
---------------	---

6.124.2.3 CopyStateToBackingFields()

```
virtual void CopyStateToBackingFields ( ) [virtual]
```

Copies the state to the backing fields. This method is meant to be overridden in derived classes.

6.124.2.4 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](#).

6.124.2.5 FixedUpdateNetwork()

```
override void FixedUpdateNetwork ( ) [virtual]
```

Fixed update callback for networked behaviours.

Reimplemented from [SimulationBehaviour](#).

6.124.2.6 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.

6.124.2.7 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.

6.124.2.8 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*

6.124.2.9 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*

6.124.2.10 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*

6.124.2.11 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.

6.124.2.12 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.

6.124.2.13 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.

6.124.2.14 GetInput< T >() [1/2]

```
T? GetInput< T > ( )
```

Template Parameters

<i>T</i>	
----------	--

Type Constraints

T : *unmanaged*

T : *INetworkInput*

6.124.2.15 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

6.124.2.16 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.

6.124.2.17 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 [LinkListReader](#) for the network linked list of type T in the network behaviour of the specified type.

6.124.2.18 GetLocalAuthorityMask()

```
int GetLocalAuthorityMask ( )
```

Gets a bitmask of [AuthorityMasks](#) flags, representing the current local authority over this [NetworkObject](#).

6.124.2.19 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**

6.124.2.20 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

6.124.2.21 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

*T*Property : *unmanaged*

6.124.2.22 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.

6.124.2.23 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.

6.124.2.24 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*

6.124.2.25 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**

6.124.2.26 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**

6.124.2.27 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**

6.124.2.28 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.

6.124.2.29 NetworkSerialize() [1/2]

```
static int NetworkSerialize (
    NetworkBehaviour obj,
    byte * data ) [static]
```

Serializes a [NetworkBehaviour](#) into a byte array.

Parameters

<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.

6.124.2.30 NetworkSerialize() [2/2]

```
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.

6.124.2.31 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.

6.124.2.32 NetworkWrap()

```
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](#).

6.124.2.33 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

6.124.2.34 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*

6.124.2.35 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.

6.124.2.36 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

6.124.2.37 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
------------------	-------------------------------------

6.124.2.38 ResetState()

```
void ResetState ( )
```

Resets the state of the object to the original state

6.124.2.39 Spawned()

```
virtual void Spawned ( ) [virtual]
```

Post spawn callback.

Implements [ISpawned](#).

Reimplemented in [NetworkTransform](#), and [NetworkMecanimAnimator](#).

6.124.2.40 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.

6.124.3 Member Data Documentation

6.124.3.1 offset

```
int offset
```

Gives access to the offset (in 32 bit words) and count (in 32 bit words) of this behaviour backing data

6.124.4 Property Documentation

6.124.4.1 ChangedTick

```
Tick ChangedTick [get]
```

The tick the data on this networked behaviour changed

6.124.4.2 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.

6.124.4.3 HasInputAuthority

```
bool? HasInputAuthority [get]
```

Returns true if the [Simulation.LocalPlayer](#) of the associated [NetworkRunner](#) is the designated as Input Source for this network entity.

6.124.4.4 HasStateAuthority

```
bool? HasStateAuthority [get]
```

Returns true if the associated [NetworkRunner](#) is the State Source for this network entity.

6.124.4.5 Id

```
NetworkBehaviourId? Id [get]
```

The unique identifier for this network behaviour.

6.124.4.6 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.

6.124.4.7 StateBuffer

```
NetworkBehaviourBuffer StateBuffer [get]
```

Gets the state buffer associated with the network behaviour.

6.124.4.8 StateBufferIsValid

```
bool StateBufferIsValid [get]
```

Gets a value indicating whether the state buffer is valid.

6.125 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.

6.125.1 Detailed Description

Provides a reader for network arrays of type T.

Template Parameters

T	The type of elements in the network array.
---	--

6.125.2 Member Function Documentation

6.125.2.1 Read()

```
NetworkArrayReadOnly<T> Read (
    NetworkBehaviourBuffer first )
```

Reads a network array from the provided network behaviour buffer.

Parameters

<code>first</code>	The network behaviour buffer to read the network array from.
--------------------	--

Returns

A read-only view of the network array.

6.126 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.

6.126.1 Detailed Description

Provides a reader for network behaviours of type T.

Template Parameters

<code>T</code>	The type of the network behaviour.
----------------	------------------------------------

Type Constraints

`T : NetworkBehaviour`

6.126.2 Member Function Documentation

6.126.2.1 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.

6.126.3 Member Data Documentation

6.126.3.1 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.

6.127 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.

6.127.1 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.

6.127.2 Member Enumeration Documentation

6.127.2.1 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.

6.127.3 Member Function Documentation

6.127.3.1 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](#).

6.127.3.2 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](#).

6.127.3.3 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.

6.128 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.

6.128.1 Detailed Description

Struct representing a collection of changes detected in a [NetworkBehaviour](#).

6.128.2 Member Function Documentation

6.128.2.1 [Changed\(\)](#)

```
bool Changed (
    string name )
```

Checks if a property has changed.

Parameters

<code>name</code>	The name of the property to check.
-------------------	------------------------------------

Returns

True if the property has changed, false otherwise.

6.128.2.2 [GetEnumerator\(\)](#)

```
Enumerator GetEnumerator ( )
```

Gets an enumerator for the collection of changes.

Returns

An [Enumerator](#) for the collection of changes.

6.129 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.

6.129.1 Detailed Description

Enumerator for the collection of changes detected in a NetworkBehaviour.

6.129.2 Member Function Documentation

6.129.2.1 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.

6.129.2.2 Reset()

```
void Reset ( )
```

Resets the enumerator to its initial state.

6.129.3 Property Documentation

6.129.3.1 Current

```
string Current [get]
```

Gets the current property name in the array of changed properties.

6.130 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.

6.130.1 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.

6.130.2 Member Function Documentation

6.130.2.1 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.

6.131 NetworkBehaviour.LinkedListReader< 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.

6.131.1 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.
----------	--

6.131.2 Member Function Documentation

6.131.2.1 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.

6.132 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.

6.132.1 Detailed Description

Provides a reader for properties of type T in a network behaviour.

Template Parameters

<code>T</code>	The type of the property in the network behaviour. Must be unmanaged.
----------------	---

Type Constraints

`T : unmanaged`

6.132.2 Constructor & Destructor Documentation

6.132.2.1 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.
---------------	---

6.132.3 Member Function Documentation

6.132.3.1 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.

6.132.4 Member Data Documentation

6.132.4.1 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.

6.133 NetworkBehaviourBuffer Struct Reference

Provides low level accesss to data buffers that can be read using a NetworkBehaviour.Reader

Public Member Functions

- float [Read \(NetworkBehaviour.PropertyReader< float > reader\)](#)
Reads a float property from the buffer using the provided PropertyReader.
- Quaternion [Read \(NetworkBehaviour.PropertyReader< Quaternion > reader\)](#)
Reads a Quaternion property from the buffer using the provided PropertyReader.
- Vector2 [Read \(NetworkBehaviour.PropertyReader< Vector2 > reader\)](#)
Reads a Vector2 property from the buffer using the provided PropertyReader.
- Vector3 [Read \(NetworkBehaviour.PropertyReader< Vector3 > reader\)](#)
Reads a Vector3 property from the buffer using the provided PropertyReader.
- Vector4 [Read \(NetworkBehaviour.PropertyReader< Vector4 > reader\)](#)
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 > \(NetworkBehaviour.PropertyReader< T > reader\)](#)
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.

6.133.1 Detailed Description

Provides low level accesss to data buffers that can be read using a NetworkBehaviour.Reader

6.133.2 Member Function Documentation

6.133.2.1 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.

6.133.2.2 Read() [1/5]

```
float Read (
    NetworkBehaviour.PropertyReader< 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.

6.133.2.3 Read() [2/5]

```
Quaternion Read (
    NetworkBehaviour.PropertyReader< 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.

6.133.2.4 Read() [3/5]

```
Vector2 Read (
    NetworkBehaviour.PropertyReader< 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.

6.133.2.5 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.

6.133.2.6 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.

6.133.2.7 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*](#)

6.133.2.8 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*](#)

6.133.2.9 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*

6.133.3 Property Documentation

6.133.3.1 Length

```
int Length [get]
```

Gets the length of the buffer.

6.133.3.2 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.

6.133.3.3 Tick

```
Tick Tick [get]
```

Gets the tick at which the buffer was created.

6.133.3.4 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.

6.134 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.

6.134.1 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.

6.134.2 Constructor & Destructor Documentation

6.134.2.1 [NetworkBehaviourBufferInterpolator\(\)](#)

```
NetworkBehaviourBufferInterpolator (
    NetworkBehaviour nb )
```

Constructor for the [NetworkBehaviourBufferInterpolator](#) struct.

Parameters

<i>nb</i>	The NetworkBehaviour instance that this interpolator is associated with.
-----------	--

6.134.3 Member Function Documentation

6.134.3.1 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.

6.134.3.2 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.

6.134.3.3 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.

6.134.3.4 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.

6.134.3.5 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.

6.134.3.6 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.

6.134.3.7 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.

6.134.3.8 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.

6.134.3.9 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.

6.134.3.10 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.

6.134.3.11 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.

6.134.3.12 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

6.134.3.13 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

6.134.3.14 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.

6.134.3.15 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.

6.134.3.16 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.

6.134.3.17 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.

6.134.3.18 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.

6.134.3.19 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.

6.134.4 Member Data Documentation

6.134.4.1 Alpha

```
readonly float Alpha
```

The interpolation factor, ranging from 0 to 1. This value is used to interpolate between the "from" and "to" states.

6.134.4.2 Behaviour

```
readonly NetworkBehaviour Behaviour
```

The [NetworkBehaviour](#) instance that this interpolator is associated with.

6.134.4.3 From

```
readonly NetworkBehaviourBuffer From
```

The [NetworkBehaviourBuffer](#) instance representing the "from" state for interpolation.

6.134.4.4 To

```
readonly NetworkBehaviourBuffer To
```

The [NetworkBehaviourBuffer](#) instance representing the "to" state for interpolation.

6.134.4.5 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.

6.135 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.

6.135.1 Detailed Description

Represents the unique identifier for a [NetworkBehaviour](#) instance.

6.135.2 Member Function Documentation

6.135.2.1 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.

6.135.2.2 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.

6.135.2.3 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](#).

6.135.2.4 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

a	The first NetworkBehaviourId to compare.
b	The second NetworkBehaviourId to compare.

Returns

True if the [NetworkBehaviourId](#) instances are not equal, false otherwise.

6.135.2.5 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

a	The first NetworkBehaviourId to compare.
b	The second NetworkBehaviourId to compare.

Returns

True if the [NetworkBehaviourId](#) instances are equal, false otherwise.

6.135.2.6 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](#).

6.135.3 Member Data Documentation

6.135.3.1 Behaviour

```
int Behaviour
```

The identifier for the behaviour within the object.

6.135.3.2 Object

```
NetworkId Object
```

The [NetworkId](#) of the object this behaviour belongs to.

6.135.3.3 SIZE

```
const int SIZE = NetworkId.SIZE + sizeof(int) [static]
```

Size of the [NetworkBehaviourId](#) structure in bytes.

6.135.4 Property Documentation

6.135.4.1 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.

6.135.4.2 None

`NetworkBehaviourId` `None` [static], [get]

Returns a new `NetworkBehaviourId` with default values.

6.136 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 [RpclinevokeData\[\]](#) delegates)
Tries to get the RPC (Remote Procedure Call) invoke delegate array for a given type.
- static bool [TryGetRpcStaticInvokeDelegate](#) (int index, out [RpclinevokeDelegate](#) 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.

6.136.1 Detailed Description

This static class provides utility methods for working with [NetworkBehaviour](#) objects.

6.136.2 Member Function Documentation

6.136.2.1 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.

6.136.2.2 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*

6.136.2.3 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 <i>D</i> is created.

Type Constraints

D : [IDictionary](#)

D : [K](#)

D : [V](#)

D : [new\(\)](#)

K : [unmanaged](#)

V : [unmanaged](#)

6.136.2.4 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**

6.136.2.5 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.

6.136.2.6 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

System.ArgumentOutOfRangeException	Thrown when the static RPC does not exist.
--	--

6.136.2.7 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.

6.136.2.8 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.
---	--

6.136.2.9 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.

6.136.2.10 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**

6.136.2.11 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

6.136.2.12 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*

6.136.2.13 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.
------------	---

6.136.2.14 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.
------------	---

6.136.2.15 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.
------------	---

6.136.2.16 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*

6.136.2.17 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.

6.136.2.18 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.

6.136.2.19 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.

6.136.2.20 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.
--------------	--------------------------------------

6.136.2.21 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.

6.136.2.22 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.

6.136.2.23 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.

6.136.2.24 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.
-------------	--

6.136.2.25 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.

6.136.2.26 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.

6.136.2.27 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.
---	---

6.136.2.28 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.

6.136.2.29 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.

6.136.3 Member Data Documentation

6.136.3.1 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.

6.137 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](#).

6.137.1 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 .
----------	--

6.137.2 Member Function Documentation

6.137.2.1 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.
---------------------------------------	---

6.137.2.2 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

<code>System.NotImplementedException</code>	Thrown always as this method is meant to be used only for [Networked] properties inline initialization.
---	---

6.138 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](#).

6.138.1 Detailed Description

A utility structure for initializing [NetworkDictionary](#) with inline initialization.

Template Parameters

<code>K</code>	The type of the keys in the NetworkDictionary .
<code>V</code>	The type of the values in the NetworkDictionary .

6.138.2 Member Function Documentation

6.138.2.1 operator NetworkDictionary< K, V >()

```
static implicit operator NetworkDictionary< K, V > (
    DictionaryInitializer< K, V > arr ) [static]
```

Implicitly converts a [DictionaryInitializer](#) to a [NetworkDictionary](#).

Parameters

<code>arr</code>	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.
---	---

6.139 NetworkBehaviourUtils.MetaData Struct Reference

This structure holds metadata for a [NetworkBehaviour](#) object.

6.139.1 Detailed Description

This structure holds metadata for a [NetworkBehaviour](#) object.

6.140 NetworkBehaviourWeavedAttribute Class Reference

Network [Behaviour](#) Weaved Attribute

Inherits Attribute.

Public Member Functions

- [NetworkBehaviourWeavedAttribute](#) (int wordCount)
NetworkBehaviourWeavedAttribute Constructor

Properties

- int [WordCount](#) [get]
NetworkBehaviour Word Count

6.140.1 Detailed Description

Network [Behaviour](#) Weaved Attribute

6.140.2 Constructor & Destructor Documentation

6.140.2.1 NetworkBehaviourWeavedAttribute()

```
NetworkBehaviourWeavedAttribute (
    int wordCount )
```

[NetworkBehaviourWeavedAttribute](#) Constructor

Parameters

<code>wordCount</code>	<code>WordCount</code>
------------------------	------------------------

6.140.3 Property Documentation

6.140.3.1 WordCount

`int WordCount [get]`

`NetworkBehaviour` Word Count

6.141 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](#)

6.141.1 Detailed Description

Represents a boolean value that can be networked.

6.141.2 Constructor & Destructor Documentation

6.141.2.1 NetworkBool()

```
NetworkBool (
    bool value )
```

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

Parameters

value	The boolean value.
-------	--------------------

6.141.3 Member Function Documentation

6.141.3.1 Equals() [1/2]

```
bool Equals (
    NetworkBool other )
```

Determines whether the specified [NetworkBool](#) is equal to the current [NetworkBool](#).

Parameters

other	The NetworkBool to compare with the current NetworkBool .
-------	---

Returns

true if the specified [NetworkBool](#) is equal to the current [NetworkBool](#); otherwise, false.

6.141.3.2 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.

6.141.3.3 GetHashCode()

```
override int GetHashCode ( )
```

Serves as the default hash function.

Returns

A hash code for the current [NetworkBool](#).

6.141.3.4 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.
------------	---

6.141.3.5 operator NetworkBool()

```
static implicit operator NetworkBool (
    bool val ) [static]
```

Defines an implicit conversion of a bool to a [NetworkBool](#).

Parameters

<i>val</i>	The bool to convert.
------------	----------------------

6.141.3.6 ToString()

```
override string ToString ( )
```

Returns a string that represents the current [NetworkBool](#).

Returns

A string that represents the current [NetworkBool](#).

6.142 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.

6.142.1 Detailed Description

Represents a set of buttons that can be networked.

6.142.2 Constructor & Destructor Documentation

6.142.2.1 `NetworkButtons()`

```
NetworkButtons (
    int buttons )
```

Initializes a new instance of the `NetworkButtons` struct with the specified buttons state.

Parameters

<code>buttons</code>	The integer representing the state of the buttons.
----------------------	--

6.142.3 Member Function Documentation

6.142.3.1 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.

6.142.3.2 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.

6.142.3.3 GetHashCode()

```
override int GetHashCode ( )
```

Serves as the default hash function.

Returns

A hash code for the current [NetworkButtons](#).

6.142.3.4 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.

6.142.3.5 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.

6.142.3.6 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.

6.142.3.7 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*

6.142.3.8 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.

6.142.3.9 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*

6.142.3.10 SetAllDown()

```
void SetAllDown ( )
```

Sets all buttons to down.

6.142.3.11 SetAllUp()

```
void SetAllUp ( )
```

Sets all buttons to up.

6.142.3.12 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.
---------------	---

6.142.3.13 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**

6.142.3.14 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.
---------------	---------------------------------------

6.142.3.15 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*

6.142.3.16 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.

6.142.3.17 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 <i>T</i> to check.

Returns

true if the button was pressed; otherwise, false.

Type Constraints

T : *unmanaged*

T : *Enum*

6.142.3.18 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.

6.142.3.19 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 <i>T</i> to check.

Returns

true if the button was released; otherwise, false.

Type Constraints

T : *unmanaged*

T : *Enum*

6.142.4 Member Data Documentation

6.142.4.1 NetworkButtons

`NetworkButtons`

Gets the buttons that were pressed or released since the previous state.

Parameters

<code>previous</code>	The previous state of the buttons.
-----------------------	------------------------------------

Returns

A tuple containing the buttons that were pressed and the buttons that were released.

6.142.5 Property Documentation

6.142.5.1 Bits

`int Bits [get]`

Gets the bits representing the state of the buttons.

6.143 NetworkConfiguration Class Reference

Main network configuration class.

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`.

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 **MtuDefault** [get]
Max number of bytes that can be used by [Fusion](#) to fill up a UDP package.
- int **SocketRecvBufferSize** [get]
Size in Kilobytes of the underlying socket receive buffer.
- int **SocketSendBufferSize** [get]
Size in Kilobytes of the underlying socket send buffer.

6.143.1 Detailed Description

Main network configuration class.

6.143.2 Member Enumeration Documentation

6.143.2.1 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.

6.143.3 Member Function Documentation

6.143.3.1 Init()

`NetworkConfiguration Init ()`

Initializes and creates a copy of this [NetworkProjectConfig](#).

Returns

A copy of this [NetworkProjectConfig](#).

6.143.4 Member Data Documentation

6.143.4.1 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).

6.143.4.2 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.

6.143.4.3 ReliableDataTransferModes

```
ReliableDataTransfers ReliableDataTransferModes
```

Initial value:

```
=  
    ReliableDataTransfers.ClientToServer |  
    ReliableDataTransfers.ClientToClientWithServerProxy
```

Current [ReliableDataTransferModes](#) mode.

6.143.5 Property Documentation

6.143.5.1 ConnectAttempts

```
int ConnectAttempts [get]
```

Max number of connection attempts that a Client will run when trying to connect to a remote Server.

6.143.5.2 ConnectInterval

```
double ConnectInterval [get]
```

Interval in seconds between each connection attempt from a Client.

6.143.5.3 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.

6.143.5.4 ConnectionPingInterval

```
double ConnectionPingInterval [get]
```

Interval in seconds between PING messages sent to a remote connection, in order to keep that connection alive.

Currently unused.

6.143.5.5 MtuDefault

```
int MtuDefault [get]
```

Max number of bytes that can be used by [Fusion](#) to fill up a UDP package.

6.143.5.6 SocketRecvBufferSize

```
int SocketRecvBufferSize [get]
```

Size in Kilobytes of the underlying socket receive buffer.

6.143.5.7 SocketSendBufferSize

```
int SocketSendBufferSize [get]
```

Size in Kilobytes of the underlying socket send buffer.

6.144 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

6.144.1 Detailed Description

Network Runner Callbacks Delegates

6.145 NetworkDeserializeMethodAttribute Class Reference

Network Deserialize Method Attribute

Inherits Attribute.

6.145.1 Detailed Description

Network Deserialize Method Attribute

6.146 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)
Returns true if the Dictionary contains an entry for the given key.
- 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)
Returns the value for the given key. Will throw an error if the key is not found.
- V [Get](#) (K key)
Returns an enumerator that iterates through the NetworkDictionary.
- IEnumarator< KeyValuePair< K, V > > [GetEnumerator](#) ()
Initializes a new instance of the NetworkDictionary struct with the specified data, capacity, and reader/writers.
- IEnumarator [IEnumerable.GetEnumerator](#) ()
- K [GetKey](#) (int entry)
Removes entry from Dictionary.
- int [GetKeyHashCode](#) (K key)
Removes entry from Dictionary. If successful (key existed), returns true and the value of removed item.
- int [GetNxt](#) (int entry)
Sets the value for the given key. Will add a new key if the key does not already exist.
- V [GetVal](#) (int entry)
Set the value for the given key. Will add a new key if the key does not already exist.
- int [Insert](#) (K key, V val)
Set the value for the given key. Will add a new key if the key does not already exist.
- [NetworkDictionary](#) (int *data, int capacity, [IElementReaderWriter< K >](#) keyReaderWriter, [IElementReaderWriter< V >](#) valReaderWriter)
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.

6.146.1 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 .

6.146.2 Constructor & Destructor Documentation**6.146.2.1 NetworkDictionary()**

```
NetworkDictionary (
    int * data,
    int capacity,
    IEElementReaderWriter< K > keyReaderWriter,
    IEElementReaderWriter< 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.

6.146.3 Member Function Documentation**6.146.3.1 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.

6.146.3.2 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](#).

6.146.3.3 Clear()

```
void Clear ( )
```

Remove all entries from the Dictionary, and clear backing memory.

6.146.3.4 ContainsKey()

```
bool ContainsKey (
    K key )
```

Returns true if the Dictionary contains an entry for the given key.

6.146.3.5 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 <code>IEqualityComparer</code> to be used for compare.

6.146.3.6 Get()

```
V Get (
    K key )
```

Returns the value for the given key. Will throw an error if the key is not found.

6.146.3.7 GetEnumerator()

```
Enumerator GetEnumerator ( )
```

Returns an enumerator that iterates through the [NetworkDictionary](#).

6.146.3.8 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](#).

6.146.3.9 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.

6.146.3.10 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.

6.146.3.11 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.

6.146.3.12 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](#).

6.146.3.13 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.

6.146.4 Member Data Documentation

6.146.4.1 META_WORD_COUNT

```
const int META_WORD_COUNT = 3 [static]
```

Meta word count for [NetworkDictionary](#).

6.146.5 Property Documentation

6.146.5.1 Capacity

```
int Capacity [get]
```

The maximum number of entries this dictionary may contain.

6.146.5.2 Count

```
int Count [get]
```

Current number of key/value entries in the Dictionary.

6.146.5.3 this[K key]

```
V this[K key] [get], [set]
```

Key indexer. Gets/Sets value for specified key.

6.147 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 \[get\]](#)

6.147.1 Detailed Description

Enumerator for [NetworkDictionary](#).

6.147.2 Member Function Documentation

6.147.2.1 Dispose()

```
void Dispose ( )
```

Dispose enumerator.

6.147.2.2 MoveNext()

```
bool MoveNext ( )
```

Move to next entry in dictionary.

Returns

Returns true if there is a next entry.

6.147.2.3 Reset()

```
void Reset ( )
```

Reset enumerator.

6.147.3 Property Documentation

6.147.3.1 Current

```
KeyValuePair<K, V> Current [get]
```

Current key/value pair.

Exceptions

<i>InvalidOperationException</i>	Thrown if enumerator is not valid.
----------------------------------	------------------------------------

6.148 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.

6.148.1 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.

6.148.2 Member Function Documentation

6.148.2.1 Get()

```
V Get (
    K key )
```

Returns the value for the given key. Will throw an error if the key is not found.

6.148.2.2 TryGet()

```
bool TryGet (
    K key,
    out V value )
```

Attempts to get the value for a given key. If found, returns true.

Parameters

<code>key</code>	The key to remove.
<code>value</code>	Returns value of removed item. Returns default value if key did not exist.

Returns

Returns true if key was found.

6.148.3 Property Documentation

6.148.3.1 Capacity

```
int Capacity [get]
```

The maximum number of entries this dictionary may contain.

6.148.3.2 Count

```
int Count [get]
```

Current number of key/value entries in the Dictionary.

6.149 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.

6.149.1 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 { } }
```

6.149.2 Constructor & Destructor Documentation

6.149.2.1 NetworkedAttribute()

```
NetworkedAttribute ()
```

Default constructor for [NetworkedAttribute](#)

6.149.3 Property Documentation

6.149.3.1 Default

```
string Default [get], [set]
```

Name of the field that holds the default value for this networked property.

6.150 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

6.150.1 Detailed Description

Networked Weaved Attribute

Networked Property Attribute

6.150.2 Constructor & Destructor Documentation

6.150.2.1 NetworkedWeavedAttribute()

```
NetworkedWeavedAttribute (
    int wordOffset,
    int wordCount )
```

NetworkedWeavedAttribute Constructor

Parameters

<code>wordOffset</code>	WordOffset
<code>wordCount</code>	WordCount

6.150.3 Property Documentation

6.150.3.1 WordCount

```
int WordCount [get]
```

Networked Property Word Count

6.150.3.2 WordOffset

```
int WordOffset [get]
```

Networked Property Word Offset

6.151 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

6.151.1 Detailed Description

Companion component for [NetworkRunner](#). Exposes [INetworkRunnerCallbacks](#) as UnityEvents, which can be wired up to other components in the inspector.

6.152 NetworkEvents.ConnectFailedEvent Class Reference

UnityEvent for ConnectFailed

Inherits UnityEvent< NetworkRunner, NetAddress, NetConnectFailedReason >.

6.152.1 Detailed Description

UnityEvent for ConnectFailed

6.153 NetworkEvents.ConnectRequestEvent Class Reference

UnityEvent for ConnectRequest

Inherits UnityEvent< NetworkRunner, NetworkRunnerCallbackArgs.ConnectRequest, byte[] >.

6.153.1 Detailed Description

UnityEvent for ConnectRequest

6.154 NetworkEvents.CustomAuthenticationResponse Class Reference

UnityEvent for Custom Authentication

Inherits UnityEvent< NetworkRunner, Dictionary< string, object >>.

6.154.1 Detailed Description

UnityEvent for Custom Authentication

6.155 NetworkEvents.DisconnectFromServerEvent Class Reference

UnityEvent for DisconnectFromServer

Inherits UnityEvent< NetworkRunner, NetDisconnectReason >.

6.155.1 Detailed Description

UnityEvent for DisconnectFromServer

6.156 NetworkEvents.HostMigrationEvent Class Reference

UnityEvent for HostMigration

Inherits UnityEvent< NetworkRunner, HostMigrationToken >.

6.156.1 Detailed Description

UnityEvent for HostMigration

6.157 NetworkEvents.InputEvent Class Reference

UnityEvent for [NetworkInput](#)

Inherits UnityEvent< NetworkRunner, NetworkInput >.

6.157.1 Detailed Description

UnityEvent for [NetworkInput](#)

6.158 NetworkEvents.InputPlayerEvent Class Reference

UnityEvent for [NetworkInput](#) with [PlayerRef](#)

Inherits UnityEvent< NetworkRunner, PlayerRef, NetworkInput >.

6.158.1 Detailed Description

UnityEvent for [NetworkInput](#) with [PlayerRef](#)

6.159 NetworkEvents.ObjectEvent Class Reference

UnityEvent for [NetworkObject](#)

Inherits UnityEvent< NetworkRunner, NetworkObject >.

6.159.1 Detailed Description

UnityEvent for [NetworkObject](#)

6.160 NetworkEvents.ObjectPlayerEvent Class Reference

UnityEvent for [NetworkObject](#) with [PlayerRef](#)

Inherits UnityEvent< NetworkRunner, NetworkObject, PlayerRef >.

6.160.1 Detailed Description

UnityEvent for [NetworkObject](#) with [PlayerRef](#)

6.161 NetworkEvents.PlayerEvent Class Reference

UnityEvent for [PlayerRef](#)

Inherits UnityEvent< NetworkRunner, PlayerRef >.

6.161.1 Detailed Description

UnityEvent for [PlayerRef](#)

6.162 NetworkEvents.ReliableDataEvent Class Reference

UnityEvent for Reliable Data

Inherits UnityEvent< NetworkRunner, PlayerRef, ReliableKey, ArraySegment< byte >>.

6.162.1 Detailed Description

UnityEvent for Reliable Data

6.163 NetworkEvents.ReliableProgressEvent Class Reference

UnityEvent for Reliable Data Progress

Inherits UnityEvent< NetworkRunner, PlayerRef, ReliableKey, float >.

6.163.1 Detailed Description

UnityEvent for Reliable Data Progress

6.164 NetworkEvents.RunnerEvent Class Reference

UnityEvent for [NetworkRunner](#)

Inherits UnityEvent< NetworkRunner >.

6.164.1 Detailed Description

UnityEvent for [NetworkRunner](#)

6.165 NetworkEvents.SessionListUpdateEvent Class Reference

UnityEvent for [SessionInfo](#) List

Inherits UnityEvent< NetworkRunner, List< SessionInfo >>.

6.165.1 Detailed Description

UnityEvent for [SessionInfo](#) List

6.166 NetworkEvents.ShutdownEvent Class Reference

UnityEvent for Shutdown

Inherits UnityEvent< NetworkRunner, ShutdownReason >.

6.166.1 Detailed Description

UnityEvent for Shutdown

6.167 NetworkEvents.SimulationMessageEvent Class Reference

UnityEvent for [SimulationMessage](#)

Inherits UnityEvent< NetworkRunner, SimulationMessagePtr >.

6.167.1 Detailed Description

UnityEvent for [SimulationMessage](#)

6.168 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.

6.168.1 Detailed Description

The unique identifier for a network entity.

6.168.2 Member Function Documentation

6.168.2.1 CompareTo()

```
int CompareTo (
    NetworkId other )
```

Compares the current **NetworkId** object with another **NetworkId** object.

Parameters

other	A NetworkId object to compare with this object.
--------------	--

Returns

A value that indicates the relative order of the objects being compared.

6.168.2.2 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.

6.168.2.3 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.

6.168.2.4 GetHashCode()

```
override int GetHashCode ( )
```

Get the hash code for this [NetworkId](#).

6.168.2.5 operator bool()

```
static implicit operator bool (
    NetworkId id ) [static]
```

Converts the `NetworkId` object to a boolean value.

6.168.2.6 operator"!=()

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

Determines whether two `NetworkId` objects are not equal.

6.168.2.7 operator==()

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

Determines whether two `NetworkId` objects are equal.

6.168.2.8 Read()

```
static NetworkId Read (
    NetBitBuffer * buffer ) [static]
```

Reads a `NetworkId` from the provided `NetBitBuffer`.

Parameters

<code>buffer</code>	The buffer to read the <code>NetworkId</code> from.
---------------------	---

Returns

The `NetworkId` read from the buffer.

6.168.2.9 ToNamePrefixString()

```
string ToNamePrefixString ( )
```

String conversion specifically for use in prefixing names of GameObjects.

6.168.2.10 `ToString()`

```
override string ToString ( )
```

String representation of the [NetworkId](#).

6.168.2.11 `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.
---------------	--

6.168.2.12 `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.

6.168.3 Member Data Documentation

6.168.3.1 ALIGNMENT

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

The alignment of the network id in bytes.

6.168.3.2 BLOCK_SIZE

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

The size of the network id block in bytes.

6.168.3.3 Raw

```
uint Raw
```

The raw value of the network id.

6.168.3.4 SIZE

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

The size of the network id in bytes.

6.168.4 Property Documentation

6.168.4.1 Comparer

```
EqualityComparer Comparer = new EqualityComparer() [static], [get]
```

The IEqualityComparer for [NetworkId](#) objects.

6.168.4.2 IsReserved

```
bool IsReserved [get]
```

Signal if the network id is reserved.

6.168.4.3 IsValid

```
bool IsValid [get]
```

Signal if the network id is valid.

6.169 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.

6.169.1 Detailed Description

IEqualityComparer interface for [NetworkId](#) objects.

6.169.2 Member Function Documentation

6.169.2.1 Equals()

```
bool Equals (
    NetworkId a,
    NetworkId b )
```

Determines whether the specified [NetworkId](#) objects are equal.

6.169.2.2 GetHashCode()

```
int GetHashCode (
    NetworkId id )
```

Returns a hash code for the specified [NetworkId](#) object.

6.170 NetworkInput Struct Reference

[NetworkInput](#) Struct

Public Member Functions

- bool `Convert` (`Type` type)
`Converts the Type of this INetworkInput to another type`
- bool `Convert< 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`

6.170.1 Detailed Description

[NetworkInput](#) Struct

6.170.2 Member Function Documentation

6.170.2.1 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

6.170.2.2 Get< T >()

```
T Get< T > ( )
```

Gets the content of this [INetworkInput](#) as another type

Type Constraints

T : unmanaged

T : INetworkInput

6.170.2.3 Is< T >()

```
bool Is< T > ( )
```

Checks if this [INetworkInput](#) is of a certain type

Type Constraints

T : unmanaged

T : INetworkInput

6.170.2.4 Set< T >()

```
bool Set< T > (
    T value )
```

Sets the content of this [INetworkInput](#) to another type

Type Constraints

T : unmanaged

T : INetworkInput

6.170.2.5 TryGet< T >()

```
bool TryGet< T > (
    out T input )
```

Tries to export data as the indicated T [INetworkInput](#) struct.

Type Constraints

T : unmanaged

T : INetworkInput

6.170.2.6 TrySet< T >()

```
bool TrySet< T > (  
    T input )
```

Tries to import data from a [INetworkInput](#) struct.

Type Constraints

T : *unmanaged*

T : [INetworkInput](#)

6.170.3 Property Documentation

6.170.3.1 Data

```
uint* Data [get]
```

Data pointer of the [NetworkInput](#)

6.170.3.2 IsValid

```
bool IsValid [get]
```

Signal if the [NetworkInput](#) is valid or not

6.170.3.3 Type

```
Type Type [get]
```

Get the Type associated with this [NetworkInput](#)

6.170.3.4 WordCount

```
int WordCount [get]
```

Number of Words for the [NetworkInput](#)

6.171 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](#)

6.171.1 Detailed Description

Utility methods for [NetworkInput](#)

6.171.2 Member Function Documentation

6.171.2.1 GetMaxWordCount()

```
static int GetMaxWordCount ( ) [static]
```

Get the max word count from all registered types

6.171.2.2 GetType()

```
static Type GetType (
    int typeKey ) [static]
```

Get the Type based on its associate Key

Parameters

<code>typeKey</code>	Key associated with a Type
----------------------	----------------------------

Returns

Type associated with the Key, or null otherwise

6.171.2.3 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

6.171.2.4 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](#)

6.172 NetworkInputWeavedAttribute Class Reference

Network Input Weaved Attribute

Inherits Attribute.

Public Member Functions

- [NetworkInputWeavedAttribute](#) (int wordCount)
NetworkInputWeavedAttribute Constructor

Properties

- int [WordCount](#) [get]
NetworkInput Word Count

6.172.1 Detailed Description

Network Input Weaved Attribute

6.172.2 Constructor & Destructor Documentation

6.172.2.1 NetworkInputWeavedAttribute()

```
NetworkInputWeavedAttribute (
    int wordCount )
```

[NetworkInputWeavedAttribute](#) Constructor

Parameters

wordCount	WordCount
---------------------------	---------------------------

6.172.3 Property Documentation

6.172.3.1 WordCount

```
int WordCount [get]
```

[NetworkInput Word Count](#)

6.173 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
Returns the number of words required to store a single element.
- const int **ELEMENT_WORDS** = 2
- 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.

6.173.1 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

<i>T</i>	<i>T</i> can be a primitive, or an INetworkStruct .
----------	---

6.173.2 Constructor & Destructor Documentation

6.173.2.1 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.

6.173.3 Member Function Documentation

6.173.3.1 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](#).

6.173.3.2 Add() [2/2]

```
void Add (
    T value )
```

Adds a value to the end of the list.

Parameters

<i>value</i>	Value to add.
--------------	---------------

6.173.3.3 Clear()

```
void Clear ( )
```

Removes and clears all list elements.

6.173.3.4 Contains() [1/2]

```
bool Contains (
    T value )
```

Returns true if the value already exists in the list.

6.173.3.5 Contains() [2/2]

```
bool Contains (
    T value,
    IEqualityComparer< T > comparer )
```

Returns true if the value already exists in the list.

6.173.3.6 Get()

```
T Get (
    int index )
```

Returns the value at supplied index.

6.173.3.7 GetEnumerator()

```
IEnumerator GetEnumerator ( )
```

Get the enumerator for the list.

6.173.3.8 IndexOf() [1/2]

```
int IndexOf (
    T value )
```

Returns the index with this value. Returns -1 if not found.

6.173.3.9 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.

6.173.3.10 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.

6.173.3.11 Remove() [1/2]

```
bool Remove (
    T value )
```

Removes the first found element with indicated value.

6.173.3.12 Remove() [2/2]

```
bool Remove (
    T value,
    IEqualityComparer< T > equalityComparer )
```

Removes the first found element with indicated value.

6.173.3.13 Set()

```
T Set (  
    int index,  
    T value )
```

Sets the value at supplied index.

6.173.4 Member Data Documentation

6.173.4.1 ELEMENT_WORDS

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

Returns the number of words required to store a single element.

6.173.4.2 META_WORDS

```
const int META_WORDS = 3 [static]
```

Returns the number of words required to store the list metadata.

6.173.5 Property Documentation

6.173.5.1 Capacity

```
int Capacity [get]
```

Returns the max element count.

6.173.5.2 Count

```
int Count [get]
```

Returns the current element count.

6.173.5.3 this[int index]

T this[int index] [get], [set]

Element indexer.

6.174 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]

6.174.1 Detailed Description

Enumerator for [NetworkLinkedList<T>](#).

6.174.2 Member Function Documentation

6.174.2.1 Dispose()

```
void Dispose ( )
```

Releases all resources used by the `NetworkLinkedList<T>.Enumerator`.

6.174.2.2 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.

6.174.2.3 Reset()

```
void Reset ( )
```

Resets the enumerator to its initial position, which is before the first element in the collection.

6.174.3 Property Documentation

6.174.3.1 Current

```
T Current [get]
```

Gets the current element in the collection.

Returns

The current element in the collection.

Exceptions

<code>InvalidOperationException</code>	Thrown when the enumerator is positioned before the first element or after the last element.
--	--

6.175 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.

6.175.1 Detailed Description

Read-only version of NetworkLinkedList<T>.

Template Parameters

<i>T</i>	Custom struct type.
----------	---------------------

6.175.2 Member Function Documentation**6.175.2.1 Contains() [1/2]**

```
bool Contains (
    T value )
```

Returns true if the value already exists in the list.

6.175.2.2 Contains() [2/2]

```
bool Contains (
    T value,
    IEqualityComparer< T > comparer )
```

Returns true if the value already exists in the list.

6.175.2.3 Get()

```
T Get (
    int index )
```

Returns the value at supplied index.

6.175.2.4 IndexOf() [1/2]

```
int IndexOf (
    T value )
```

Returns the index with this value. Returns -1 if not found.

6.175.2.5 IndexOf() [2/2]

```
int IndexOf (
    T value,
    IEqualityComparer< T > equalityComparer )
```

Returns the index of the first occurrence of a value in the [FixedArray](#).

Parameters

<code>value</code>	The value to locate in the FixedArray . The value can be null for reference types.
<code>equalityComparer</code>	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.

6.175.3 Member Data Documentation

6.175.3.1 ELEMENT_WORDS

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

Returns the number of words required to store a single element.

6.175.3.2 META_WORDS

```
const int META_WORDS = 3 [static]
```

Returns the number of words required to store the list metadata.

6.175.4 Property Documentation

6.175.4.1 Capacity

```
int Capacity [get]
```

Returns the max element count.

6.175.4.2 Count

```
int Count [get]
```

Returns the current element count.

6.175.4.3 this[int index]

`T this[int index] [get]`

Element indexer.

6.176 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

6.176.1 Detailed Description

Parameters for loading a scene

6.176.2 Member Function Documentation

6.176.2.1 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

6.176.2.2 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

6.176.2.3 GetHashCode()

```
override int GetHashCode ( )
```

Returns the hash code of the [NetworkLoadSceneParameters](#)

6.176.2.4 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

6.176.2.5 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

6.176.2.6 ToString()

```
override string ToString ( )
```

Returns a string representation of the [NetworkLoadSceneParameters](#)

6.176.3 Member Data Documentation

6.176.3.1 LoadId

```
readonly NetworkSceneLoadId LoadId
```

The unique id of the scene load operation

6.176.4 Property Documentation

6.176.4.1 IsActiveOnLoad

```
bool IsActiveOnLoad [get]
```

Signals if the scene should be active on load

6.176.4.2 IsLocalPhysics2D

```
bool IsLocalPhysics2D [get]
```

Signals if the scene should have local 2D physics

6.176.4.3 IsLocalPhysics3D

```
bool IsLocalPhysics3D [get]
```

Signals if the scene should have local 3D physics

6.176.4.4 IsSingleLoad

```
bool IsSingleLoad [get]
```

Signals if the scene should be single loaded

6.176.4.5 LoadSceneMode

```
LoadSceneMode LoadSceneMode [get]
```

The [LoadSceneMode](#) to use when loading the scene

6.176.4.6 LoadSceneParameters

LoadSceneParameters LoadSceneParameters [get]

The [LoadSceneParameters](#) to use when loading the scene

6.176.4.7 LocalPhysicsMode

LocalPhysicsMode LocalPhysicsMode [get]

The [LocalPhysicsMode](#) to use when loading the scene

6.177 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

6.177.1 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.

6.177.2 Member Function Documentation

6.177.2.1 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

<code>triggerHash</code>	Trigger hash to set
<code>passThroughOnInputAuthority</code>	Will call <code>Animator.SetTrigger()</code> immediately on the InputAuthority. If false, SetTrigger() will not be called on the Input Authority at all and <code>Animator.SetTrigger()</code> should be called explicitly as needed.

6.177.2.2 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 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.

6.177.3 Member Data Documentation**6.177.3.1 Animator**

```
Animator Animator
```

The Animator being synced. If unset, will attempt to find one on this GameObject.

6.177.3.2 ApplyTiming

```
RenderSource ApplyTiming = RenderSource.To
```

The source of the State which is applied in Render.

6.177.4 Property Documentation**6.177.4.1 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

System.InvalidOperationException	Thrown when this property is accessed outside of playing.
--	---

6.178 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 its [NetworkId](#).

Public Attributes

- **NetworkObjectFlags Flags**
Flags used for network object prefabs and similar
- **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 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]**

- string **Name** [get]

Last tick this object received an update.
- **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.

6.178.1 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.

6.178.2 Member Function Documentation

6.178.2.1 AssignInputAuthority()

```
void AssignInputAuthority (
    PlayerRef player )
```

Sets which [PlayerRef](#) has Input Authority for this Object.

6.178.2.2 Awake()

```
virtual void Awake ( ) [protected], [virtual]
```

Awake is called when the script instance is being loaded.

6.178.2.3 CopyStateFrom() [1/2]

```
void CopyStateFrom (
    NetworkObject source )
```

Copies the entire State from another [NetworkObject](#)

Parameters

<i>source</i>	NetworkObject to copy the State from
---------------	--

6.178.2.4 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
---------------	---

6.178.2.5 GetLocalAuthorityMask()

```
int GetLocalAuthorityMask ( )
```

Gets a bitmask of [AuthorityMasks](#) flags, representing the current local authority over this [NetworkObject](#).

6.178.2.6 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

System.Exception	Thrown when a NetworkBehaviour reference is missing in the NetworkBehaviour array of the NetworkObject .
----------------------------------	--

6.178.2.7 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

6.178.2.8 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)

6.178.2.9 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)

6.178.2.10 OnDestroy()

```
virtual void OnDestroy ( ) [protected], [virtual]
```

OnDestroy is called when the script instance is being destroyed.

6.178.2.11 operator NetworkId()

```
static implicit operator NetworkId (
    NetworkObject obj ) [static]
```

Converts the Network Object to it's [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)

6.178.2.12 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.

6.178.2.13 ReleaseStateAuthority()

```
void ReleaseStateAuthority ( )
```

Release the state authority over this [NetworkObject](#) on shared mode.

6.178.2.14 RemoveInputAuthority()

```
void RemoveInputAuthority ( )
```

Removes input authority from whichever player has it for this object. Only valid when called on a Host or Server peer.

6.178.2.15 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.

6.178.2.16 RequestStateAuthority()

```
void RequestStateAuthority ( )
```

Request state authority over this [NetworkObject](#) on shared mode.

6.178.2.17 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

6.178.3 Member Data Documentation

6.178.3.1 Flags

`NetworkObjectFlags Flags`

Flags used for network object prefabs and similar

6.178.3.2 IsResume

`bool IsResume`

Signal that this `NetworkObject` comes from a Resume Spawn

6.178.3.3 NestedObjects

`NetworkObject [] NestedObjects`

Array of initial child nested `NetworkObject` entities, that are children of this Object.

6.178.3.4 NetworkedBehaviours

`NetworkBehaviour [] NetworkedBehaviours`

Array of all `NetworkBehaviour`s associated with this network entity.

6.178.3.5 NetworkTypeId

`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.

6.178.3.6 PriorityCallback

```
PriorityLevelDelegate PriorityCallback
```

Delegate callback used to override priority value for a specific object-player pair

6.178.3.7 ReplicateTo

```
ReplicateToDelegate ReplicateTo
```

Delegate callback used to override if an object should be replicate to a client or not

6.178.3.8 SortKey

```
uint SortKey
```

Used for whenever objects need to be sorted in a deterministic order, like when registering scene objects.

6.178.4 Property Documentation

6.178.4.1 HasInputAuthority

```
bool HasInputAuthority [get]
```

Returns if [Simulation.LocalPlayer](#) is the designated Input Source for this network entity.

6.178.4.2 HasStateAuthority

```
bool HasStateAuthority [get]
```

Returns if [Simulation.LocalPlayer](#) is the designated State Source for this network entity.

6.178.4.3 Id

```
NetworkId? Id [get]
```

The unique identifier for this network entity.

6.178.4.4 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.

6.178.4.5 IsInSimulation

```
bool IsInSimulation [get]
```

If this object is inserted into the simulation

6.178.4.6 IsProxy

```
bool IsProxy [get]
```

Returns if [Simulation.LocalPlayer](#) is neither the Input nor State Source for this network entity.

6.178.4.7 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.

6.178.4.8 IsValid

```
bool IsValid [get]
```

Returns if this network entity is associated with its [NetworkRunner](#), and that runner is not null.

6.178.4.9 LastReceiveTick

```
Tick LastReceiveTick [get]
```

Last tick this object received an update.

6.178.4.10 Name

```
string Name [get]
```

The ID + Unity GameObject name for this entity.

6.178.4.11 RenderSource

```
RenderSource RenderSource [get], [set]
```

Returns the [Fusion.RenderSource](#) for this [Fusion.NetworkBehaviour](#) instance, indicating how snapshot data will be used to render it.

6.178.4.12 RenderTime

```
float RenderTime [get]
```

Returns the current interpolation time for this object

6.178.4.13 RenderTimeframe

```
RenderTimeframe RenderTimeframe [get]
```

Returns the [Fusion.RenderTimeframe](#) for this [Fusion.NetworkBehaviour](#) instance, indicating what snapshot data will be used to render it.

6.178.4.14 Runner

```
NetworkRunner Runner [get]
```

The [NetworkRunner](#) this entity is associated with.

6.178.4.15 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.

6.179 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.

6.179.1 Detailed Description

Extension methods for the NetworkObjectFlags enum.

6.179.2 Member Function Documentation

6.179.2.1 GetVersion()

```
static int GetVersion (
    this NetworkObjectFlags flags ) [static]
```

Returns the version of the flags.

Parameters

<code>flags</code>	The flags to get the version of.
--------------------	----------------------------------

Returns

The version of the flags.

6.179.2.2 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.

6.179.2.3 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.

6.179.2.4 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.

6.179.2.5 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.

6.180 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.

6.180.1 Detailed Description

NetworkObjectGuid

6.180.2 Constructor & Destructor Documentation

6.180.2.1 NetworkObjectGuid() [1/4]

```
NetworkObjectGuid (
    string guid )
```

Create a new NetworkObjectGuid

Parameters

<i>guid</i>	The guid to use
-------------	-----------------

6.180.2.2 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

6.180.2.3 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
-------------	---

6.180.2.4 NetworkObjectGuid() [4/4]

```
NetworkObjectGuid (
    byte * guid )
```

Create a [NetworkObjectGuid](#) from a byte*

Parameters

<i>guid</i>	The byte* to create the NetworkObjectGuid from
-------------	--

6.180.3 Member Function Documentation

6.180.3.1 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

6.180.3.2 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 NetworkObjectGuids are equal, false otherwise

6.180.3.3 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

6.180.3.4 GetHashCode()

```
override int GetHashCode ( )
```

Get the hashcode for a [NetworkObjectGuid](#)

6.180.3.5 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

6.180.3.6 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](#)

6.180.3.7 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](#)

6.180.3.8 operator"!=()

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

Compare two [NetworkObjectGuid](#)

Returns

True if the [NetworkObjectGuid](#) are not equal, false otherwise

6.180.3.9 operator==()

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

Compare two [NetworkObjectGuid](#)

Returns

True if the [NetworkObjectGuid](#) are equal, false otherwise

6.180.3.10 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](#).

6.180.3.11 ToString() [1/2]

```
override string ToString ( )
```

Returns a string representation of the [NetworkObjectGuid](#).

6.180.3.12 ToString() [2/2]

```
string ToString (
    string format )
```

Returns a string representation of the [NetworkObjectGuid](#).

6.180.3.13 ToUnityGuidString()

```
string ToUnityGuidString ( )
```

Returns a string representation of the [NetworkObjectGuid](#).

6.180.3.14 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

6.180.4 Member Data Documentation

6.180.4.1 ALIGNMENT

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

The alignment of the [NetworkObjectGuid](#)

6.180.4.2 RawGuidValue

```
fixed long RawGuidValue[2]
```

The Raw Guid Value of the [NetworkObjectGuid](#)

6.180.4.3 SIZE

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

The Size of the [NetworkObjectGuid](#) in bytes

6.180.5 Property Documentation

6.180.5.1 Empty

```
NetworkObjectGuid Empty [static], [get]
```

The default value of a [NetworkObjectGuid](#)

6.180.5.2 IsValid

```
bool IsValid [get]
```

Signal if the [NetworkObjectGuid](#) is valid.

6.181 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

6.181.1 Detailed Description

EqualityComparer for NetworkObjectGuid

6.181.2 Member Function Documentation

6.181.2.1 Equals()

```
bool Equals (
    NetworkObjectGuid x,
    NetworkObjectGuid y )
```

Check if two NetworkObjectGuid are equals

6.181.2.2 GetHashCode()

```
int GetHashCode (
    NetworkObjectGuid obj )
```

Get the hashcode for a NetworkObjectGuid

6.182 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 number of words in the network object header.

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]
how many bytes this headers object is

6.182.1 Detailed Description

Network object header information for a `NetworkObject`.

6.182.2 Member Function Documentation

6.182.2.1 Equals() [1/2]

```
bool Equals (
    NetworkObjectHeader other )
```

Checks if the current instance of `NetworkObjectHeader` is equal to another instance of the same type.

6.182.2.2 Equals() [2/2]

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

Checks if the current instance of `NetworkObjectHeader` is equal to another object.

6.182.2.3 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](#).

6.182.2.4 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](#).

6.182.2.5 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](#).

6.182.2.6 GetHashCode()

```
override int GetHashCode ( )
```

Generates a hash code for the current instance of [NetworkObjectHeader](#).

6.182.2.7 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.

6.182.2.8 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.

6.182.2.9 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.

6.182.2.10 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.

6.182.2.11 ToString()

```
override string ToString ( )
```

The string representation of the [NetworkObjectHeader](#).

6.182.3 Member Data Documentation

6.182.3.1 _reserved

```
fixed int _reserved[10]
```

Reserved space for future use.

6.182.3.2 BehaviourCount

```
short BehaviourCount
```

The number of behaviours in the network object.

6.182.3.3 Flags

```
NetworkObjectHeaderFlags Flags
```

The flags indicating various states or properties of the network object.

6.182.3.4 Id

`NetworkId` `Id`

The unique identifier of the network object.

6.182.3.5 InputAuthority

`PlayerRef` `InputAuthority`

The player reference who has input authority over the network object.

6.182.3.6 NestingKey

`NetworkObjectNestingKey` `NestingKey`

The nesting key of the network object.

6.182.3.7 NestingRoot

`NetworkId` `NestingRoot`

The unique identifier of the root network object in the nesting hierarchy.

6.182.3.8 PLAYER_DATA_WORD

`const int` `PLAYER_DATA_WORD` $= 36 / \text{Allocator.REPLICATE_WORD_SIZE}$ [static]

The word index of the player data in the `NetworkObjectHeader`.

6.182.3.9 SIZE

`const int` `SIZE` $= 80$ [static]

The size of the `NetworkObjectHeader` in bytes.

6.182.3.10 StateAuthority

`PlayerRef` StateAuthority

The player reference who has state authority over the network object.

6.182.3.11 Type

`NetworkObjectTypeId` Type

The type identifier of the network object.

6.182.3.12 WordCount

`short` WordCount

The number of words in the network object header.

6.182.3.13 WORDS

`const int WORDS = SIZE / Allocator.REPLICATE_WORD_SIZE [static]`

The size of the `NetworkObjectHeader` in words.

6.182.4 Property Documentation

6.182.4.1 ByteCount

`int ByteCount [get]`

how many bytes this headers object is

6.183 NetworkObjectHeaderPtr Struct Reference

Represents a pointer to a `NetworkObjectHeader`. This struct is unsafe because it uses pointers.

Public Attributes

- `NetworkObjectHeader * Ptr`
Pointer to a [NetworkObjectHeader](#).

Properties

- `NetworkId Id [get]`
Gets the Id of the [NetworkObjectHeader](#) this struct points to.
- `NetworkObjectTypeID Type [get]`
Gets the Type of the [NetworkObjectHeader](#) this struct points to.

6.183.1 Detailed Description

Represents a pointer to a [NetworkObjectHeader](#). This struct is unsafe because it uses pointers.

6.183.2 Member Data Documentation

6.183.2.1 Ptr

`NetworkObjectHeader* Ptr`

Pointer to a [NetworkObjectHeader](#).

6.183.3 Property Documentation

6.183.3.1 Id

`NetworkId Id [get]`

Gets the Id of the [NetworkObjectHeader](#) this struct points to.

6.183.3.2 Type

`NetworkObjectTypeID Type [get]`

Gets the Type of the [NetworkObjectHeader](#) this struct points to.

6.184 NetworkObjectInitializerUnity Class Reference

Initializes network objects for Unity.

Inherits [INetworkObjectInitializer](#).

Public Member Functions

- void [InitializeNetworkState](#) ([NetworkObject](#) networkObject)
Initializes the network object.

6.184.1 Detailed Description

Initializes network objects for Unity.

6.184.2 Member Function Documentation

6.184.2.1 InitializeNetworkState()

```
void InitializeNetworkState (
    NetworkObject networkObject )
```

Initializes the network object.

Parameters

networkObject	The network object to initialize.
-------------------------------	-----------------------------------

Implements [INetworkObjectInitializer](#).

6.185 NetworkObjectMeta Class Reference

Meta information about a network object.

Properties

- [NetworkId Id](#) [get]
Get the [NetworkId](#) of this object.
- [PlayerRef InputAuthority](#) [get]
Get the Player that has input authority over this object.
- [PlayerRef StateAuthority](#) [get]
Get the Player that has state authority over this object.
- [NetworkObjectTypeId Type](#) [get]
Get the [NetworkObjectTypeId](#) of this object.

6.185.1 Detailed Description

Meta information about a network object.

6.185.2 Property Documentation

6.185.2.1 Id

`NetworkId` `Id` [get]

Get the `NetworkId` of this object.

6.185.2.2 InputAuthority

`PlayerRef` `InputAuthority` [get]

Get the Player that has input authority over this object.

6.185.2.3 StateAuthority

`PlayerRef` `StateAuthority` [get]

Get the Player that has state authority over this object.

6.185.2.4 Type

`NetworkObjectType` `Type` [get]

Get the `NetworkObjectType` of this object.

6.186 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.

6.186.1 Detailed Description

A key used to identify a network object nesting.

6.186.2 Constructor & Destructor Documentation

6.186.2.1 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 .
--------------	--

6.186.3 Member Function Documentation

6.186.3.1 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.

6.186.3.2 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.

6.186.3.3 GetHashCode()

```
override int GetHashCode ( )
```

Serves as the default hash function.

Returns

A hash code for the current object.

6.186.3.4 ToString()

```
override string ToString ( )
```

Returns a string that represents the current object.

Returns

A string that represents the current object.

6.186.4 Member Data Documentation**6.186.4.1 ALIGNMENT**

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

The alignment of the [NetworkObjectNestingKey](#) in bytes.

6.186.4.2 SIZE

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

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

6.186.4.3 Value

```
int Value
```

The value of the [NetworkObjectNestingKey](#).

6.186.5 Property Documentation

6.186.5.1 IsNone

```
bool IsNone [get]
```

Checks if the [NetworkObjectNestingKey](#) is none.

Returns

True if the value of the [NetworkObjectNestingKey](#) is 0; otherwise, false.

6.186.5.2 IsValid

```
bool IsValid [get]
```

Checks if the [NetworkObjectNestingKey](#) is valid.

Returns

True if the value of the [NetworkObjectNestingKey](#) is greater than 0; otherwise, false.

6.187 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](#).

6.187.1 Detailed Description

Implements the [IEqualityComparer](#) interface.

6.187.2 Member Function Documentation

6.187.2.1 Equals()

```
bool Equals (
    NetworkObjectNestingKey x,
    NetworkObjectNestingKey y )
```

Determines whether two [NetworkObjectNestingKey](#) objects are equal.

6.187.2.2 GetHashCode()

```
int GetHashCode (
    NetworkObjectNestingKey obj )
```

Returns a hash code for the specified [NetworkObjectNestingKey](#).

6.188 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

6.188.1 Detailed Description

This class represents the data for a network object prefab.

6.188.2 Member Data Documentation

6.188.2.1 Guid

[NetworkObjectGuid](#) Guid

The unique identifier for the network object.

6.189 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.
- `void ReleaseInstance (NetworkRunner runner, in NetworkObjectReleaseContext context)`
Releases an instance of a network object.

6.189.1 Detailed Description

A dummy implementation of the [INetworkObjectProvider](#) interface. This class is used for testing purposes and throws a NotImplementedException for all its methods.

6.190 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.

6.190.1 Detailed Description

Represents the context for releasing a network object. This struct is unsafe because it uses pointers.

6.190.2 Constructor & Destructor Documentation

6.190.2.1 NetworkObjectReleaseContext()

```
NetworkObjectReleaseContext (
    NetworkObject obj,
    NetworkObjectType typeId,
    bool isBeingDestroyed,
    bool isNested )
```

Initializes a new instance of the `NetworkObjectReleaseContext` struct with the specified parameters.

Parameters

<code>obj</code>	The network object to be released.
<code>typeId</code>	The type identifier of the network object.
<code>isBeingDestroyed</code>	Indicates whether the network object is being destroyed.
<code>isNested</code>	Indicates whether the network object is a nested object.

6.190.3 Member Function Documentation

6.190.3.1 ToString()

```
override string ToString ( )
```

Returns a string that represents the current object.

Returns

A string that represents the current object.

6.190.4 Member Data Documentation

6.190.4.1 IsBeingDestroyed

```
readonly bool IsBeingDestroyed
```

Indicates whether the network object is being destroyed.

6.190.4.2 IsNestedObject

```
readonly bool IsNestedObject
```

Indicates whether the network object is a nested object.

6.190.4.3 Object

```
readonly NetworkObject Object
```

The network object to be released.

6.190.4.4 TypeId

```
readonly NetworkObjectType Id TypeId
```

The type identifier of the network object.

6.191 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.

6.191.1 Detailed Description

This class is used to compare two [NetworkObject](#) instances based on their SortKey. It implements the IComparer interface.

6.191.2 Member Function Documentation

6.191.2.1 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*.

6.191.3 Member Data Documentation**6.191.3.1 Instance**

```
readonly NetworkObjectSortKeyComparer Instance = new NetworkObjectSortKeyComparer\(\) [static]
```

An instance of the [NetworkObjectSortKeyComparer](#) class.

6.192 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
- [NetworkObjectTypeId?](#) [Typeld](#) [get]
Network Object Type Id

6.192.1 Detailed Description

Network Object Spawn Exception

6.192.2 Constructor & Destructor Documentation**6.192.2.1 NetworkObjectSpawnException()**

```
NetworkObjectSpawnException (
    NetworkSpawnStatus status,
    NetworkObjectTypeId? id = null )
```

Network Object Spawn Exception Constructor

Parameters

<code>status</code>	Network Spawn Status
<code>id</code>	Network Object Type Id

6.192.3 Property Documentation

6.192.3.1 Message

```
override string Message [get]
```

Exception Message

6.192.3.2 Status

```
NetworkSpawnStatus Status [get]
```

Network Spawn Status

6.192.3.3 Typeld

```
NetworkObjectType? Typeld [get]
```

Network Object Type Id

6.193 NetworkObjectTypeld Struct Reference

ID for a [NetworkObject](#) Prefab which has been cataloged in a [NetworkProjectConfig.PrefabTable](#).

Inherits [INetworkStruct](#), and [IEquatable<NetworkObjectTypeld>](#).

Classes

- class [EqualityComparer](#)
NetworkObjectTypeld Comparer

Public Member Functions

- bool [Equals \(NetworkObjectTypeld other\)](#)
Checks if the current `NetworkObjectTypeld` instance is equal to another `NetworkObjectTypeld` instance.
- override bool [Equals \(object obj\)](#)
Determines whether the specified object is equal to the current `NetworkObjectTypeld` instance.
- override int [GetHashCode \(\)](#)
Generates a hash code for the current `NetworkObjectTypeld` instance.
- override string [ToString \(\)](#)
Returns a string that represents the current `NetworkObjectTypeld` instance.

Static Public Member Functions

- static [NetworkObjectTypeld FromCustom \(uint raw\)](#)
Creates a `NetworkObjectTypeld` from a raw uint value representing a Custom type.
- static [NetworkObjectTypeld FromPrefabId \(NetworkPrefabId prefabId\)](#)
Creates a `NetworkObjectTypeld` from a `NetworkPrefabId`.
- static [NetworkObjectTypeld FromSceneRefAndObjectIndex \(SceneRef sceneRef, int objIndex, NetworkSceneLoadId loadId=default\)](#)
Creates a `NetworkObjectTypeld` from a `SceneRef`, an object index, and an optional `NetworkSceneLoadId`.
- static [NetworkObjectTypeld FromStruct \(ushort structId\)](#)
Creates a `NetworkObjectTypeld` from a ushort value representing an InternalStruct type.
- static implicit operator [NetworkObjectTypeld \(NetworkPrefabId prefabId\)](#)
Converts a `NetworkPrefabId` instance to a `NetworkObjectTypeld` instance.
- static bool [operator!= \(NetworkObjectTypeld a, NetworkObjectTypeld b\)](#)
Determines whether two `NetworkObjectTypeld` instances are not equal.
- static bool [operator== \(NetworkObjectTypeld a, NetworkObjectTypeld b\)](#)
Determines whether two `NetworkObjectTypeld` instances are equal.

Public Attributes

- uint [_value0](#)
Represents the first part of the value of a `NetworkObjectTypeld`.
- uint [_value1](#)
Represents the second part of the value of a `NetworkObjectTypeld`.

Static Public Attributes

- const int [ALIGNMENT = 4](#)
Represents the alignment of a `NetworkObjectTypeld` 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 `NetworkObjectTypeld`.
- const int [SIZE = 8](#)
Represents the size of a `NetworkObjectTypeld` 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.

6.193.1 Detailed Description

ID for a `NetworkObject` Prefab which has been cataloged in a `NetworkProjectConfig.PrefabTable`.

6.193.2 Member Function Documentation

6.193.2.1 Equals() [1/2]

```
bool Equals (
    NetworkObjectTypeId other )
```

Checks if the current `NetworkObjectTypeId` instance is equal to another `NetworkObjectTypeId` instance.

Parameters

<code>other</code>	The other <code>NetworkObjectTypeId</code> instance to compare with the current instance.
--------------------	---

6.193.2.2 Equals() [2/2]

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

Determines whether the specified object is equal to the current [NetworkObjectTypeId](#) instance.

Parameters

<i>obj</i>	The object to compare with the current NetworkObjectTypeId instance.
------------	--

6.193.2.3 FromCustom()

```
static NetworkObjectTypeId FromCustom (
    uint raw ) [static]
```

Creates a [NetworkObjectTypeId](#) from a raw uint value representing a Custom type.

Parameters

<i>raw</i>	The raw uint value to use.
------------	----------------------------

Returns

A [NetworkObjectTypeId](#) that represents a Custom type with the given raw value.

6.193.2.4 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.
--------------------------	--

6.193.2.5 FromSceneRefAndObjectIndex()

```
static NetworkObjectTypeId FromSceneRefAndObjectIndex (
    SceneRef sceneRef,
    int objIndex,
    NetworkSceneLoadId loadId = default ) [static]
```

Creates a [NetworkObjectTypeld](#) 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 [NetworkObjectTypeld](#) 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.

6.193.2.6 FromStruct()

```
static NetworkObjectTypeId FromStruct (
    ushort structId ) [static]
```

Creates a [NetworkObjectTypeld](#) from a ushort value representing an InternalStruct type.

Parameters

<i>struct</i> ↪ <i>Id</i>	The ushort value to use.
------------------------------	--------------------------

Returns

A [NetworkObjectTypeId](#) that represents an InternalStruct type with the given ushort value.

6.193.2.7 GetHashCode()

```
override int GetHashCode ( )
```

Generates a hash code for the current [NetworkObjectTypeId](#) instance.

6.193.2.8 operator NetworkObjectTypeId()

```
static implicit operator NetworkObjectTypeId (
    NetworkPrefabId prefabId ) [static]
```

Converts a [NetworkPrefabId](#) instance to a [NetworkObjectTypeId](#) instance.

Parameters

<i>prefabId</i>	The NetworkPrefabId instance to convert.
-----------------	--

Returns

A [NetworkObjectTypeId](#) instance that represents a Prefab with the given [NetworkPrefabId](#).

6.193.2.9 operator"!=()

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

Determines whether two [NetworkObjectTypeId](#) instances are not equal.

6.193.2.10 operator==()

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

Determines whether two [NetworkObjectTypeId](#) instances are equal.

6.193.2.11 ToString()

```
override string ToString ( )
```

Returns a string that represents the current [NetworkObjectTypeld](#) instance.

6.193.3 Member Data Documentation

6.193.3.1 _value0

```
uint _value0
```

Represents the first part of the value of a [NetworkObjectTypeld](#).

6.193.3.2 _value1

```
uint _value1
```

Represents the second part of the value of a [NetworkObjectTypeld](#).

6.193.3.3 ALIGNMENT

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

Represents the alignment of a [NetworkObjectTypeld](#) in memory.

6.193.3.4 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 [NetworkObjectTypeld](#).

6.193.3.5 SIZE

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

Represents the size of a [NetworkObjectTypeld](#) in bytes.

6.193.4 Property Documentation

6.193.4.1 AsCustom

```
uint AsCustom [get]
```

Gets the raw uint value representation of the [NetworkObjectTypeld](#) assuming it is a Custom type.

The raw uint value representation of the [NetworkObjectTypeld](#).

Exceptions

<i>InvalidOperationException</i>	Thrown when the NetworkObjectTypeld is not a Custom type.
----------------------------------	---

6.193.4.2 AsInternalStructId

```
ushort AsInternalStructId [get]
```

Gets the ushort value representation of the [NetworkObjectTypeld](#) assuming it is an InternalStruct type.

The ushort value representation of the [NetworkObjectTypeld](#).

Exceptions

<i>InvalidOperationException</i>	Thrown when the NetworkObjectTypeld is not an InternalStruct type.
----------------------------------	--

6.193.4.3 AsPrefabId

```
NetworkPrefabId AsPrefabId [get]
```

Gets the [NetworkPrefabId](#) representation of the [NetworkObjectTypeld](#) assuming it is a Prefab.

The [NetworkPrefabId](#) representation of the [NetworkObjectTypeld](#).

Exceptions

<i>InvalidOperationException</i>	Thrown when the NetworkObjectTypeld is not a Prefab.
----------------------------------	--

6.193.4.4 AsSceneObjectId

```
NetworkSceneObjectId AsSceneObjectId [get]
```

Gets the [NetworkSceneObjectId](#) representation of the [NetworkObjectTypeld](#) assuming it is a SceneObject.

The [NetworkSceneObjectId](#) representation of the [NetworkObjectTypeld](#).

Exceptions

<i>InvalidOperationException</i>	Thrown when the NetworkObjectTypeld is not a SceneObject.
----------------------------------	---

6.193.4.5 Comparer

```
EqualityComparer Comparer = new EqualityComparer() [static], [get]
```

An instance of the [NetworkObjectTypeld EqualityComparer](#) class.

6.193.4.6 IsCustom

```
bool IsCustom [get]
```

Checks if the [NetworkObjectTypeld](#) is a Custom type.

6.193.4.7 IsNone

```
bool IsNone [get]
```

Checks if the [NetworkObjectTypeld](#) is invalid.

6.193.4.8 IsPrefab

```
bool IsPrefab [get]
```

Checks if the [NetworkObjectTypeld](#) is a Prefab.

6.193.4.9 IsSceneObject

```
bool IsSceneObject [get]
```

Checks if the [NetworkObjectTypeld](#) is a SceneObject.

6.193.4.10 IsStruct

```
bool IsStruct [get]
```

Checks if the [NetworkObjectTypeld](#) is an InternalStruct.

6.193.4.11 IsValid

```
bool IsValid [get]
```

Checks if the [NetworkObjectTypeId](#) is valid.

6.193.4.12 Kind

```
NetworkTypeIdKind Kind [get]
```

Gets the kind of the [NetworkObjectTypeId](#).

6.193.4.13 PlayerData

```
NetworkObjectTypeId PlayerData [static], [get]
```

Represents a [NetworkObjectTypeId](#) for the PlayerData.

6.194 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.

6.194.1 Detailed Description

[NetworkObjectTypeId](#) Comparer

6.194.2 Member Function Documentation

6.194.2.1 Equals()

```
bool Equals (
    NetworkObjectTypeId x,
    NetworkObjectTypeId y )
```

Checks if two `NetworkObjectTypeId` instances are equal.

6.194.2.2 GetHashCode()

```
int GetHashCode (
    NetworkObjectTypeId obj )
```

Gets the hash code of a `NetworkObjectTypeId` instance.

Parameters

<i>obj</i>	The NetworkObjectTypeld instance.
------------	---

6.195 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

6.195.1 Detailed Description

Network Physics [INetworkStruct](#)

6.195.2 Member Data Documentation

6.195.2.1 SIZE

```
const int SIZE = 40 [static]
```

[NetworkPhysicsInfo](#) Total Size

6.195.2.2 TimeScale

```
float TimeScale
```

[NetworkPhysicsInfo](#) Time Scale

6.195.2.3 WORD_COUNT

```
const int WORD_COUNT = 10 [static]
```

[NetworkPhysicsInfo](#) Word Count

6.196 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 isSyncronous=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 [IsSyncronous](#)
Indicates whether the operation is syncronous.
- 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.

6.196.1 Detailed Description

Represents the context for acquiring a prefab instance for a network object. This struct is unsafe because it uses pointers.

6.196.2 Constructor & Destructor Documentation

6.196.2.1 NetworkPrefabAcquireContext()

```
NetworkPrefabAcquireContext (
    NetworkPrefabId prefabId,
    NetworkObjectMeta meta = null,
    bool isSyncronous = 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.

6.196.3 Member Data Documentation

6.196.3.1 DontDestroyOnLoad

```
readonly bool DontDestroyOnLoad
```

Indicates whether the network object should not be destroyed on load.

6.196.3.2 IsSynchronous

```
readonly bool IsSynchronous
```

Indicates whether the operation is synchronous.

6.196.3.3 Meta

```
readonly NetworkObjectMeta Meta
```

The metadata of the network object.

6.196.3.4 PrefabId

```
readonly NetworkPrefabId PrefabId
```

The identifier of the prefab.

6.196.4 Property Documentation

6.196.4.1 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

<i>InvalidOperationException</i>	Thrown when the Header is null.
----------------------------------	---------------------------------

6.196.4.2 HasHeader

bool HasHeader [get]

Checks if the Header is not null.

Returns

True if the Header is not null; otherwise, false.

6.197 NetworkPrefabAttribute Class Reference

Network Prefab Attribute

Inherits PropertyAttribute.

6.197.1 Detailed Description

Network Prefab Attribute

6.198 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.

6.198.1 Detailed Description

ID for a [NetworkObject](#) Prefab which has been cataloged in a [NetworkProjectConfig.PrefabTable](#).

6.198.2 Member Function Documentation

6.198.2.1 CompareTo() [1/2]

```
int CompareTo (
    NetworkPrefabId other )
```

Compares the [NetworkPrefabId](#) to another [NetworkPrefabId](#).

6.198.2.2 CompareTo() [2/2]

```
int IComparable.CompareTo (
    object obj )
```

Compares the [NetworkPrefabId](#) to another object.

6.198.2.3 Equals() [1/2]

```
bool Equals (
    NetworkPrefabId other )
```

Checks if the [NetworkPrefabId](#) is equal to another [NetworkPrefabId](#).

6.198.2.4 Equals() [2/2]

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

Checks if the [NetworkPrefabId](#) is equal to another object.

6.198.2.5 FromIndex()

```
static NetworkPrefabId FromIndex (
    int index ) [static]
```

Creates a [NetworkPrefabId](#) from an index.

6.198.2.6 FromRaw()

```
static NetworkPrefabId FromRaw (
    uint value ) [static]
```

Creates a [NetworkPrefabId](#) from a raw value.

6.198.2.7 GetHashCode()

```
override int GetHashCode ( )
```

Gets the hash code of the [NetworkPrefabId](#).

6.198.2.8 operator"!=()

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

Checks if two [NetworkPrefabId](#) are not equal.

6.198.2.9 operator==()

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

Checks if two [NetworkPrefabId](#) are equal.

6.198.2.10 ToString() [1/2]

```
override string ToString ( )
```

Converts the [NetworkPrefabId](#) to a string.

6.198.2.11 ToString() [2/2]

```
string ToString (
    bool brackets,
    bool prefix )
```

Converts the [NetworkPrefabId](#) to a string with optional brackets and prefix.

6.198.3 Member Data Documentation

6.198.3.1 ALIGNMENT

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

The alignment of a [NetworkPrefabId](#).

6.198.3.2 MAX_INDEX

```
const int MAX_INDEX = int.MaxValue - 1 [static]
```

The maximum index value of a [NetworkPrefabId](#).

6.198.3.3 RawValue

```
uint RawValue
```

The raw value of the [NetworkPrefabId](#).

6.198.3.4 SIZE

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

The size of a [NetworkPrefabId](#).

6.198.4 Property Documentation

6.198.4.1 AsIndex

```
int AsIndex [get]
```

Converts the [NetworkPrefabId](#) to an index.

6.198.4.2 IsNone

```
bool IsNone [get]
```

Checks if the [NetworkPrefabId](#) is none.

6.198.4.3 IsValid

```
bool IsValid [get]
```

Checks if the [NetworkPrefabId](#) is valid.

6.199 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](#).

6.199.1 Detailed Description

Equality comparer for [NetworkPrefabId](#).

6.199.2 Member Function Documentation

6.199.2.1 Equals()

```
bool Equals (
    NetworkPrefabId x,
    NetworkPrefabId y )
```

Checks if two `NetworkPrefabId` are equal.

6.199.2.2 GetHashCode()

```
int GetHashCode (
    NetworkPrefabId obj )
```

Gets the hash code of a `NetworkPrefabId`.

6.200 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.

6.200.1 Detailed Description

Meta data for a `NetworkObject` prefab which has been cataloged in a `NetworkProjectConfig.PrefabTable`.

6.200.2 Member Data Documentation

6.200.2.1 Header

```
readonly NetworkObjectHeader* Header
```

Header data for the [NetworkObject](#) prefab.

6.200.2.2 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](#).

6.200.2.3 Prefab

```
readonly NetworkPrefabId Prefab
```

Prefab ID. Use [NetworkPrefabTable.TryAddSource](#) to look up the actual prefab reference in the [NetworkProjectConfig.PrefabTable](#).

6.200.3 Property Documentation

6.200.3.1 Data

```
int* Data [get]
```

Data pointer to the first word of this [NetworkObject](#)'s data block.

6.200.3.2 HasHeader

```
bool HasHeader [get]
```

If the Header is not null.

6.201 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.

6.201.1 Detailed Description

[NetworkPrefabRef](#)

A decoupled [NetworkObject](#) prefab reference. Internally stored as a GUID.

6.201.2 Constructor & Destructor Documentation

6.201.2.1 [NetworkPrefabRef\(\)](#) [1/4]

```
NetworkPrefabRef (
    string guid )
```

Create a new [NetworkPrefabRef](#)

Parameters

<code>guid</code>	The guid to use
-------------------	-----------------

6.201.2.2 [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

6.201.2.3 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
-------------	--

6.201.2.4 NetworkPrefabRef() [4/4]

```
NetworkPrefabRef (   
    byte * guid )
```

Create a [NetworkPrefabRef](#) from a byte*

Parameters

<i>guid</i>	The byte* to create the NetworkPrefabRef from
-------------	---

6.201.3 Member Function Documentation

6.201.3.1 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

6.201.3.2 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

6.201.3.3 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

6.201.3.4 GetHashCode()

```
override int GetHashCode ( )
```

Get the hashCode for a [NetworkPrefabRef](#)

6.201.3.5 operator Guid()

```
static implicit operator Guid (
    NetworkPrefabRef guid ) [static]
```

Implicit conversion from [NetworkPrefabRef](#) to Guid

Parameters

guid	NetworkPrefabRef to convert from
------	--

Returns

Guid

6.201.3.6 operator NetworkObjectGuid()

```
static operator NetworkObjectGuid (
    NetworkPrefabRef t ) [explicit], [static]
```

Explicit conversion from [NetworkPrefabRef](#) to NetworkObjectGuid

Parameters

t	NetworkPrefabRef to convert from
---	--

Returns

[NetworkObjectGuid](#)

6.201.3.7 operator NetworkPrefabRef()

```
static implicit operator NetworkPrefabRef (
    Guid guid ) [static]
```

Implicit conversion from Guid to [NetworkPrefabRef](#)

Parameters

guid	Guid to convert from
------	----------------------

Returns

[NetworkPrefabRef](#)

6.201.3.8 operator"!=()

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

Compare two [NetworkPrefabRef](#)

Returns

True if the [NetworkPrefabRef](#) are not equal, false otherwise

6.201.3.9 operator==()

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

Compare two [NetworkPrefabRef](#)

Returns

True if the [NetworkPrefabRef](#) are equal, false otherwise

6.201.3.10 Parse()

```
static NetworkPrefabRef Parse (
    string str ) [static]
```

Parse a [NetworkPrefabRef](#) from a string.

Parameters

<code>str</code>	The string to parse.
------------------	----------------------

Returns

The parsed [NetworkPrefabRef](#).

6.201.3.11 ToString() [1/2]

```
override string ToString ( )
```

Returns a string representation of the [NetworkPrefabRef](#).

6.201.3.12 ToString() [2/2]

```
string ToString (
    string format )
```

Returns a string representation of the [NetworkPrefabRef](#).

6.201.3.13 ToUnityGuidString()

```
string ToUnityGuidString ( )
```

Returns a string representation of the [NetworkPrefabRef](#).

6.201.3.14 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

6.201.4 Member Data Documentation

6.201.4.1 ALIGNMENT

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

The alignment of the [NetworkPrefabRef](#)

6.201.4.2 RawGuidValue

```
fixed long RawGuidValue[2]
```

The Raw Guid Value of the [NetworkPrefabRef](#)

6.201.4.3 SIZE

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

The Size of the [NetworkPrefabRef](#) in bytes

6.201.5 Property Documentation

6.201.5.1 Empty

```
NetworkPrefabRef Empty [static], [get]
```

The default value of a [NetworkPrefabRef](#)

6.201.5.2 IsValid

```
bool IsValid [get]
```

Signal if the [NetworkPrefabRef](#) is valid.

6.202 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

6.202.1 Detailed Description

EqualityComparer for NetworkPrefabRef

6.202.2 Member Function Documentation

6.202.2.1 Equals()

```
bool Equals (
    NetworkPrefabRef x,
    NetworkPrefabRef y )
```

Check if two NetworkPrefabRef are equals

6.202.2.2 GetHashCode()

```
int GetHashCode (
    NetworkPrefabRef obj )
```

Get the hashCode for a NetworkPrefabRef

6.203 NetworkPrefabTable Class Reference

Class representing a table of network prefabs.

Public Member Functions

- int [AddInstance \(NetworkPrefabId prefabId\)](#)
Add an instance of a prefab id.
- void [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.

6.203.1 Detailed Description

Class representing a table of network prefabs.

6.203.2 Member Function Documentation

6.203.2.1 AddInstance()

```
int AddInstance (
    NetworkPrefabId prefabId )
```

Add an instance of a prefab id.

Parameters

<i>prefab</i> ↗ <i>Id</i>	Id of the prefab.
------------------------------	-------------------

Returns

The new instance count, or 0 if not found.

6.203.2.2 AddSource()

```
void 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.
--------------------------	--

6.203.2.3 Clear()

```
void Clear ( )
```

Clear the prefab table.

6.203.2.4 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.

6.203.2.5 GetEntries()

```
IEnumerable<(NetworkPrefabId, INetworkPrefabSource)> GetEntries ( )
```

Returns all entries in the table.

6.203.2.6 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.

6.203.2.7 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.

6.203.2.8 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.

6.203.2.9 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.

6.203.2.10 GetSource() [2/2]

```
INetworkPrefabSource GetSource (
    NetworkPrefabId prefabId )
```

Gets a prefab source by id.

Parameters

<i>prefab</i> <i>Id</i>	Id of the prefab source.
----------------------------	--------------------------

Returns

The prefab source, or default if not found.

6.203.2.11 IsAcquired()

```
bool IsAcquired (
    NetworkPrefabId prefabId )
```

Signal if a prefab id has been acquired.

Parameters

<i>prefab</i> <i>Id</i>	Id of the prefab.
----------------------------	-------------------

Returns

True if the prefab id has been acquired.

6.203.2.12 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.

6.203.2.13 RemoveInstance()

```
int RemoveInstance (
    NetworkPrefabId prefabId )
```

Remove an instance of a prefab id.

Parameters

<i>prefabId</i>	Id of the prefab.
-----------------	-------------------

Returns

The new instance count, or 0 if not found.

6.203.2.14 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.
------------------------------	----------------------------------

6.203.2.15 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.

6.203.2.16 UnloadAll()

```
void UnloadAll ( )
```

Unload all prefabs.

6.203.2.17 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.

6.203.3 Member Data Documentation

6.203.3.1 Options

```
NetworkPrefabTableOptions Options = NetworkPrefabTableOptions.Default
```

Options for the [NetworkPrefabTable](#).

6.203.4 Property Documentation

6.203.4.1 Prefabs

```
IReadOnlyList<INetworkPrefabSource> Prefabs [get]
```

All prefab sources.

6.203.4.2 Version

```
int Version [get]
```

Prefab table version. Incremented every time a change occurs.

6.204 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.

6.204.1 Detailed Description

Options for the [NetworkPrefabTable](#).

6.204.2 Member Data Documentation

6.204.2.1 Default

```
NetworkPrefabTableOptions Default [static]
```

Initial value:

```
= new NetworkPrefabTableOptions() {
    UnloadPrefabOnReleasingLastInstance = false,
    UnloadUnusedPrefabsOnShutdown = true,
}
```

Default options.

6.204.2.2 UnloadPrefabOnReleasingLastInstance

```
bool UnloadPrefabOnReleasingLastInstance
```

If true, prefabs will be unloaded when the last instance is released.

6.204.2.3 UnloadUnusedPrefabsOnShutdown

```
bool UnloadUnusedPrefabsOnShutdown
```

If true, all prefabs will be unloaded on shutdown.

6.205 NetworkProjectConfig Class Reference

The core [Fusion](#) config file that is shared with all peers at startup.

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.
- 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

- 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 [NetworkIdIsObjectName](#)

- 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.

6.205.1 Detailed Description

The core **Fusion** config file that is shared with all peers at startup.

6.205.2 Member Enumeration Documentation

6.205.2.1 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.

6.205.2.2 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).

6.205.3 Member Function Documentation**6.205.3.1 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

6.205.3.2 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.

6.205.3.3 Serialize()

```
static string Serialize (
    NetworkProjectConfig config ) [static]
```

Serialize a [NetworkProjectConfig](#) into a JSON string.

Parameters

<i>config</i>	NetworkProjectConfig reference
---------------	--

Returns

JSON String

6.205.3.4 ToString()

```
override string ToString ( )
```

[ToString\(\)](#) implementation.

6.205.3.5 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.

6.205.4 Member Data Documentation

6.205.4.1 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.

6.205.4.2 BuildTypes

```
NetworkRunner. BuildTypes [static]
```

Get the version information for the Fusion.Runntime.dll.

6.205.4.3 CheckNetworkedPropertiesBeingEmpty

```
bool CheckNetworkedPropertiesBeingEmpty = false
```

If set, the weaver will check if [NetworkedAttribute](#) properties getters and setters are empty.

6.205.4.4 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.

6.205.4.5 CurrentTypeId

```
const string CurrentTypeId = nameof(NetworkProjectConfig) [static]
```

Current [NetworkProjectConfig](#) Type ID

6.205.4.6 CurrentVersion

```
const int CurrentVersion = 1 [static]
```

Current [NetworkProjectConfig](#) version

6.205.4.7 DefaultResourceName

```
const string DefaultResourceName = nameof(NetworkProjectConfig) [static]
```

Default file name for the [NetworkProjectConfig](#) asset

6.205.4.8 EncryptionConfig

```
EncryptionConfig EncryptionConfig = new EncryptionConfig()
```

Reference to [EncryptionConfig](#) settings for this [NetworkProjectConfig](#)

6.205.4.9 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.

6.205.4.10 Heap

```
HeapConfiguration Heap = new HeapConfiguration()
```

Heap Settings

6.205.4.11 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.

6.205.4.12 HostMigration

```
HostMigrationConfig HostMigration = new HostMigrationConfig()
```

Reference to [HostMigration](#) settings for this [NetworkProjectConfig](#).

6.205.4.13 InvokeRenderInBatchMode

```
bool InvokeRenderInBatchMode = true
```

Signal if the [SimulationBehaviour.Render](#) callbacks should be invoked in Batch Mode.

6.205.4.14 LagCompensation

```
LagCompensationSettings LagCompensation = new LagCompensationSettings()
```

Advanced lag compensation buffer settings.

6.205.4.15 Network

```
NetworkConfiguration Network = new NetworkConfiguration()
```

Reference to [NetworkConfiguration](#) settings for this [NetworkProjectConfig](#).

6.205.4.16 NetworkConditions

```
NetworkSimulationConfiguration NetworkConditions = new NetworkSimulationConfiguration()
```

Settings for simulating network conditions of latency and loss.

6.205.4.17 NetworkIdIsObjectName

```
bool NetworkIdIsObjectName
```

Signal if the [NetworkId](#) of the [NetworkObject](#) should be included on the name of the GameObject.

6.205.4.18 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.

6.205.4.19 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).

6.205.4.20 PrefabTable

```
NetworkPrefabTable PrefabTable = new NetworkPrefabTable()
```

Reference to the [NetworkPrefabTable](#) instance for this [NetworkProjectConfig](#).

6.205.4.21 Simulation

```
SimulationConfig Simulation = new SimulationConfig()
```

Reference to [SimulationConfig](#) settings for this [NetworkProjectConfig](#).

6.205.4.22 TimeSynchronizationOverride

```
TimeSyncConfiguration TimeSynchronizationOverride
```

this can be used to override the time synchronization from code

6.205.4.23 TypeId

```
string TypeId = CurrentTypeId
```

Current [NetworkProjectConfig](#) Type ID

6.205.4.24 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.

6.205.4.25 Version

```
int Version = CurrentVersion
```

Current [NetworkProjectConfig](#) version

6.205.5 Property Documentation

6.205.5.1 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](#).

6.206 NetworkProjectConfigAsset Class Reference

Manages and references the current instance of [NetworkProjectConfig](#)

Inherits [FusionGlobalScriptableObject](#)< T >.

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 be registered at runtime with `NetworkPrefabTable.TryAddSource`.*

Properties

- `static NetworkProjectConfigAsset Global [get]`
The current `NetworkProjectConfig` instance.
- `static bool IsGlobalLoaded [get]`
True if the `NetworkProjectConfig` instance exists, otherwise false.

6.206.1 Detailed Description

Manages and references the current instance of `NetworkProjectConfig`

6.206.2 Member Function Documentation

6.206.2.1 TryGetGlobal()

```
static bool TryGetGlobal (
    out NetworkProjectConfigAsset global ) [static]
```

Try to get the current `NetworkProjectConfig` instance.

Parameters

<code>global</code>	<code>NetworkProjectConfig</code> instance if it exists, otherwise null.
---------------------	--

Returns

True if the `NetworkProjectConfig` instance exists, otherwise false.

6.206.2.2 UnloadGlobal()

```
static void UnloadGlobal ( ) [static]
```

Unload the current [NetworkProjectConfig](#) instance.

6.206.3 Member Data Documentation

6.206.3.1 BehaviourMeta

```
SerializableSimulationBehaviourMeta [ ] BehaviourMeta = Array.Empty<SerializableSimulationBehaviourMeta>()
```

An auto-generated list containing meta information about all the [SimulationBehaviours](#) in the project, e.g. execution order.

6.206.3.2 Config

```
NetworkProjectConfig Config = new NetworkProjectConfig()
```

The current [NetworkProjectConfig](#) instance.

6.206.3.3 PrefabOptions

```
NetworkPrefabTableOptions PrefabOptions = NetworkPrefabTableOptions.Default
```

Options for the [NetworkPrefabTable](#).

6.206.3.4 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 registered at runtime with [NetworkPrefabTable.TryAddSource](#).

6.206.4 Property Documentation

6.206.4.1 Global

```
NetworkProjectConfigAsset Global [static], [get]
```

The current [NetworkProjectConfig](#) instance.

6.206.4.2 IsGlobalLoaded

```
bool IsGlobalLoaded [static], [get]
```

True if the [NetworkProjectConfig](#) instance exists, otherwise false.

6.207 NetworkProjectConfigAsset.SerializableSimulationBehaviourMeta Struct Reference

An auto-generated list containing meta information about all the [SimulationBehaviour](#)s 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](#).

6.207.1 Detailed Description

An auto-generated list containing meta information about all the [SimulationBehaviour](#)s in the project, e.g. execution order.

6.207.2 Member Data Documentation

6.207.2.1 ExecutionOrder

```
int ExecutionOrder
```

The execution order of the [SimulationBehaviour](#).

6.207.2.2 Type

`SerializableType<SimulationBehaviour> Type`

The type of the [SimulationBehaviour](#).

6.208 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.

6.208.1 Detailed Description

PCG32 random generator, 16 bytes in size. <http://www.pcg-random.org>

6.208.2 Constructor & Destructor Documentation

6.208.2.1 NetworkRNG()

```
NetworkRNG ( Int32 seed )
```

Creates a new instance of **NetworkRNG** with a random seed.

Parameters

<i>seed</i>	Seed value.
-------------	-------------

6.208.3 Member Function Documentation

6.208.3.1 Next()

```
double Next ( )
```

Generates a random double value within the inclusive range [0, 1].

Returns

A random double value between 0 and 1, inclusive.

6.208.3.2 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).

6.208.3.3 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.

6.208.3.4 NextSingle()

```
float NextSingle ( )
```

Generates a random float value within the inclusive range [0, 1].

Returns

A random float value between 0 and 1, inclusive.

6.208.3.5 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).

6.208.3.6 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.

6.208.3.7 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.

6.208.3.8 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.

6.208.3.9 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.

6.208.3.10 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.

6.208.3.11 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.

6.208.3.12 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.

6.208.3.13 ToString()

```
override string ToString ( )
```

String representation of the RNG state.

6.208.4 Member Data Documentation

6.208.4.1 MAX

```
const UInt32 MAX = UInt32.MaxValue [static]
```

Maximum allowed value

6.208.4.2 SIZE

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

Size of the struct in bytes.

6.208.5 Property Documentation

6.208.5.1 Peek

```
NetworkRNG Peek [get]
```

Returns the same RNG instance.

6.209 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

6.209.1 Detailed Description

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

6.209.2 Constructor & Destructor Documentation

6.209.2.1 NetworkRpcStaticWeavedInvokerAttribute()

```
NetworkRpcStaticWeavedInvokerAttribute (
    string key )
```

NetworkRpcStaticWeavedInvokerAttribute Constructor

Parameters

<i>key</i>	Key
------------	---------------------

6.209.3 Property Documentation

6.209.3.1 Key

string [Key](#) [get]

RPC Key

6.210 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

6.210.1 Detailed Description

Network Rpc Weaved Invoker Attribute Contains info about a weaved RPC Method

6.210.2 Constructor & Destructor Documentation

6.210.2.1 NetworkRpcWeavedInvokerAttribute()

```
NetworkRpcWeavedInvokerAttribute (
    int key,
    int sources,
    int targets )
```

[NetworkRpcWeavedInvokerAttribute](#) Constructor

Parameters

<i>key</i>	Key
<i>sources</i>	Sources
<i>targets</i>	Targets

6.210.3 Property Documentation

6.210.3.1 Key

int Key [get]

RPC Key

6.210.3.2 Sources

int Sources [get]

RPC Sources

6.210.3.3 Targets

int Targets [get]

RPC Targets

6.211 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
- 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 **EnsureRunnerScenesActive** (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.
- 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.*
- 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 [IsValid \(PlayerRef player\)](#)
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? useDefaultCloudPorts=false, CancellationToken cancellationToken=default, bool useCachedRegions=false)`

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 setActiveOnLoad=DefaultSetActiveOnLoad)`

Loads a scene
- `NetworkSceneAsyncOp LoadScene (SceneRef sceneRef, LoadSceneParameters parameters, bool setActiveOnLoad=DefaultSetActiveOnLoad)`
- `NetworkSceneAsyncOp LoadScene (string sceneName, LoadSceneMode loadSceneMode=Load< SceneMode.Single, LocalPhysicsMode localPhysicsMode=LocalPhysicsMode.None, bool setActiveOnLoad=DefaultSetActiveOnLoad)`

Loads a scene
- `NetworkSceneAsyncOp LoadScene (string sceneName, LoadSceneParameters parameters, bool setActiveOnLoad=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 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 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)`

- **SetAreaOfInterest** (*int width*, *int height*)

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.
- 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
- 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 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)`

- TryGetSceneInfo (out NetworkSceneInfo sceneInfo)

Try to gets the [NetworkObject](#) associated with a specific player
- TrySetPhysicsInfo (NetworkPhysicsInfo info)

Tries to get the [NetworkSceneInfo](#) of this [NetworkRunner](#).
- 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.
- 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.
- 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.
- 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.
- 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.
- 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 List< NetworkRunner >.Enumerator GetInstancesEnumerator ()

Get enumerator for the collection of all [NetworkRunner](#)s. 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

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 IReadOnlyList< 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 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 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. 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.
- **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 0).
- **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](#).
- **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 0).
- **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 0).
- **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

6.211.1 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](#).

6.211.2 Member Enumeration Documentation

6.211.2.1 BuildTypes

enum `BuildTypes` [strong]

Enumeration of Fusion.Runtime.dll options.

Enumerator

Debug	Use the Debug version of the Fusion.Runtime.dll.
Release	Use the Release version of the Fusion.Runtime.dll.

6.211.2.2 States

enum `States` [strong]

Initialization stages of [Fusion](#)

Enumerator

Starting	Runner is about to start
Running	Runner is running
Shutdown	Runner is shutdown

6.211.3 Member Function Documentation

6.211.3.1 AddCallbacks()

```
void AddCallbacks (
    params INetworkRunnerCallbacks[ ] callbacks )
```

Register an [INetworkRunnerCallbacks](#) instance for callbacks from this [NetworkRunner](#).

Parameters

<i>callbacks</i>	Callbacks to register
------------------	-----------------------

6.211.3.2 AddGlobal()

```
void AddGlobal (
    SimulationBehaviour instance )
```

Add and register a [SimulationBehaviour](#) to this [NetworkRunner](#). Note: It should NOT be a [NetworkBehaviour](#)

6.211.3.3 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.

6.211.3.4 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

ArgumentNullException	Thrown when the provided network object is null.
ArgumentException	Thrown when the provided network object has an invalid NetworkTypeid.

6.211.3.5 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.

6.211.3.6 ClearPlayerAreaOfInterest()

```
void ClearPlayerAreaOfInterest (
    PlayerRef player )
```

Clears the area of interest for a player. This can only be called from the server/host

6.211.3.7 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.
----------------------------------	---

6.211.3.8 DestroySingleton< T >()

```
void DestroySingleton< T > ( )
```

Removes a specific [SimulationBehaviour](#) from this [NetworkRunner](#) gameobject, if it exists.

Type Constraints

T : SimulationBehaviour

6.211.3.9 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.

6.211.3.10 EnsureRunnerScenesActive()

```
bool EnsureRunnerSceneIsActive (
    out Scene previousActiveScene )
```

Ensures the scene of this runner is active and returns the previous active scene

Parameters

<code>previousActiveScene</code>	Previous active scene
----------------------------------	-----------------------

Returns

True if the scene was changed, false otherwise

6.211.3.11 Exists() [1/2]

```
bool Exists (
    NetworkId id )
```

Returns if the [Fusion.Simulation](#) contains a [NetworkObject](#) with given *id* in the current State.

6.211.3.12 Exists() [2/2]

```
bool Exists (
    NetworkObject obj )
```

Returns if the [Fusion.Simulation](#) contains a reference to a [NetworkObject](#) in the current State.

6.211.3.13 FindObject()

```
NetworkObject FindObject (
    NetworkId networkId )
```

Get the [NetworkObject](#) instance for this [NetworkRunner](#) from a [NetworkId](#).

Parameters

<code>networkId</code>	NetworkID to look forward
------------------------	---------------------------

Returns

null if object cannot be found.

6.211.3.14 GetAllBehaviours()

```
SimulationBehaviour [ ] GetAllBehaviours (
    Type type )
```

Returns an array of all [SimulationBehaviour](#) instances registered with this [NetworkRunner](#).

Parameters

<code>type</code>	The type of the behaviours to be returned.
-------------------	--

Returns

An array of [SimulationBehaviour](#) instances of the specified type.

6.211.3.15 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

<code>T</code>	SimulationBehaviour type
----------------	--

Returns

The result list

Type Constraints

`T : SimulationBehaviour`

6.211.3.16 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

<code>result</code>	The list to add the behaviours
---------------------	--------------------------------

Template Parameters

<i>T</i>	SimulationBehaviour type
----------	--------------------------

Type Constraints

T : SimulationBehaviour

6.211.3.17 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.
---------------	--

6.211.3.18 GetInputForPlayer< T >()

```
T? GetInputForPlayer< T > (
    PlayerRef player )
```

Returns the [NetworkInput](#) data from player, converted to the indicated [INetworkInput](#).

Type Constraints

T : unmanaged

T : INetworkInput

6.211.3.19 GetInstancesEnumerator()

```
static List<NetworkRunner>.Enumerator GetInstancesEnumerator ( ) [static]
```

Get enumerator for the collection of all [NetworkRunners](#). Allows to enumerate alloc-free.

6.211.3.20 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

6.211.3.21 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

6.211.3.22 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

6.211.3.23 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

6.211.3.24 GetPhysicsScene()

```
PhysicsScene GetPhysicsScene ( )
```

Get the 3D Physics scene being used by this Runner.

6.211.3.25 GetPhysicsScene2D()

```
PhysicsScene2D GetPhysicsScene2D ( )
```

Get the 2D Physics scene being used by this Runner.

6.211.3.26 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.

6.211.3.27 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

6.211.3.28 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](#)

6.211.3.29 GetPlayerObject()

```
NetworkObject GetPlayerObject (
    PlayerRef player )
```

Gets the network object associated with a specific player

Parameters

<i>player</i>	<code>PlayerRef</code> to get the network object
---------------	--

Returns

Network object if one is associated with the player

6.211.3.30 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
------------------	---

6.211.3.31 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>	<code>PlayerRef</code> to get the UserID. If no <code>PlayerRef</code> is passed, the UserID of the local client is returned instead.
---------------	---

Returns

UserID if valid player found, otherwise null.

6.211.3.32 GetRawInputForPlayer()

```
NetworkInput? GetRawInputForPlayer (
    PlayerRef player )
```

Returns the unconverted unsafe `NetworkInput` for the indicated player.

6.211.3.33 GetResumeSnapshotNetworkObjects()

```
IEnumerable<NetworkObject> GetResumeSnapshotNetworkObjects ( )
```

Iterate over the old NetworkObjects from the Resume Snapshot

Returns

Iterable list of [NetworkObject](#)

6.211.3.34 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

6.211.3.35 GetRpcTargetStatus()

```
RpcTargetStatus GetRpcTargetStatus ( PlayerRef target )
```

Return the [RpcTargetStatus](#) for a specific player.

6.211.3.36 GetRunnerForObject()

```
static NetworkRunner GetRunnerForObject ( GameObject gameObject ) [static]
```

Get the [NetworkRunner](#) a GameObject instance belongs to.

Parameters

<code>gameObject</code>	GameObject to check for a NetworkRunner
-------------------------	---

Returns

[NetworkRunner](#) reference, or null if not found

6.211.3.37 GetRunnerForScene()

```
static NetworkRunner GetRunnerForScene (
    Scene scene ) [static]
```

Get the [NetworkRunner](#) from a specific Scene

Parameters

scene	Scene to check for a NetworkRunner
-------	--

Returns

[NetworkRunner](#) reference, or null if not found

6.211.3.38 GetSingleton< T >()

```
T GetSingleton< T > ( )
```

Ensures that a specific [SimulationBehaviour](#) component exists on this [NetworkRunner](#) gameobject.

Type Constraints

T : SimulationBehaviour

6.211.3.39 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

6.211.3.40 InstantiateInRunnerScene() [1/2]

```
GameObject InstantiateInRunnerScene (
    GameObject original )
```

Instantiates an object in the scene of this runner

6.211.3.41 InstantiateInRunnerScene() [2/2]

```
GameObject InstantiateInRunnerScene (
    GameObject original,
    Vector3 position,
    Quaternion rotation )
```

Instantiates an object in the scene of this runner

6.211.3.42 InstantiateInRunnerScene< T >() [1/2]

```
T InstantiateInRunnerScene< T > (
    T original )
```

Instantiates an object in the scene of this runner

Type Constraints

T : Component

6.211.3.43 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

6.211.3.44 InvokeSceneLoadDone()

```
void InvokeSceneLoadDone (
    in SceneLoadDoneArgs info )
```

Invoke [INetworkRunnerCallbacks.OnSceneLoadDone\(NetworkRunner\)](#) on all implementations

6.211.3.45 InvokeSceneLoadStart()

```
void InvokeSceneLoadStart (
    SceneRef sceneRef )
```

Invoke [INetworkRunnerCallbacks.OnSceneLoadStart\(NetworkRunner\)](#) on all implementations

6.211.3.46 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)

6.211.3.47 IsValid()

```
bool IsValid (
    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.

6.211.3.48 JoinSessionLobby()

```
async Task<StartGameResult> JoinSessionLobby (
    SessionLobby sessionLobby,
    string lobbyID = null,
    AuthenticationValues authentication = null,
    FusionAppSettings customAppSettings = null,
    bool? useDefaultCloudPorts = false,
    CancellationToken cancellationToken = default,
    bool useCachedRegions = false )
```

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
<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.

6.211.3.49 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

6.211.3.50 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

6.211.3.51 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

6.211.3.52 MakeDontDestroyOnLoad()

```
void MakeDontDestroyOnLoad (
    GameObject obj )
```

Mark an object as DontDestroyOnLoad.

Parameters

<i>obj</i>	Object to mark
------------	----------------

6.211.3.53 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

6.211.3.54 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

6.211.3.55 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

6.211.3.56 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*

6.211.3.57 ObjectDelegate()

```
delegate void ObjectDelegate (
    NetworkRunner runner,
    NetworkObject obj )
```

Delegate type for object callback

6.211.3.58 OnBeforeSpawned()

```
delegate void OnBeforeSpawned (
    NetworkRunner runner,
    NetworkObject obj )
```

Delegate type for on before spawned callback

6.211.3.59 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.

6.211.3.60 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

ArgumentException	Thrown when the provided scene is not valid.
ArgumentNullException	Thrown when the provided objects array is null.

6.211.3.61 RemoveCallbacks()

```
void RemoveCallbacks (
    params INetworkRunnerCallbacks[ ] callbacks )
```

Unregister an [INetworkRunnerCallbacks](#) instance for callbacks from this [NetworkRunner](#).

Parameters

<i>callbacks</i>	Callbacks to unregister
------------------	-------------------------

6.211.3.62 RemoveGlobal()

```
void RemoveGlobal (
    SimulationBehaviour instance )
```

Removes a specific [SimulationBehaviour](#) from this [NetworkObject](#) gameobject, if it exists.

6.211.3.63 RenderInternal()

```
void RenderInternal ( )
```

This method is meant to be called by [INetworkRunnerUpdater](#).

6.211.3.64 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.

6.211.3.65 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.

6.211.3.66 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
----------------	---------------------------

6.211.3.67 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

6.211.3.68 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
------------------	---------------------------------------

6.211.3.69 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
------------------	---------------------------------------

6.211.3.70 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

6.211.3.71 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

6.211.3.72 SetIsSimulated()

```
bool SetIsSimulated (
    NetworkObject obj,
    bool simulate )
```

Sets the simulation state for this object, if it takes part in the NetworkFixedUpdate, etc.

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

6.211.3.73 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
---------------	--

6.211.3.74 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.

6.211.3.75 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

6.211.3.76 Shutdown()

```
Task Shutdown (
    bool destroyGameObject = true,
    ShutdownReason shutdownReason = ShutdownReason.Ok,
    bool forceShutdownProcedure = false )
```

Initiates a [Simulation.Dispose](#).

6.211.3.77 SinglePlayerContinue()

```
void SinglePlayerContinue ( )
```

Continues a paused game in single player

6.211.3.78 SinglePlayerPause() [1/2]

```
void SinglePlayerPause ( )
```

Pauses the game in single player

6.211.3.79 SinglePlayerPause() [2/2]

```
void SinglePlayerPause (
    bool paused )
```

Sets the paused state in a single player

6.211.3.80 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

6.211.3.81 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

6.211.3.82 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

6.211.3.83 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

6.211.3.84 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

6.211.3.85 `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](#)

6.211.3.86 `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 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

,

Parameters

<i>onCompleted</i>	A callback to fire once the spawn is done.
--------------------	--

6.211.3.87 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.
--------------------	--

6.211.3.88 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
<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.
--------------------	--

6.211.3.89 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.
--------------------	--

6.211.3.90 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.
--------------------	--

6.211.3.91 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](#)

6.211.3.92 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

6.211.3.93 TryFindBehaviour()

```
bool TryFindBehaviour (
    NetworkBehaviourId behaviourId,
    out NetworkBehaviour behaviour )
```

Get the [NetworkBehaviour](#) instance for this [NetworkRunner](#) from a [NetworkBehaviourId](#).

Parameters

<i>behaviourId</i>	NetworkBehaviourId to look forward
<i>behaviour</i>	NetworkBehaviour reference, if found

Returns

True if object was found.

6.211.3.94 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*

6.211.3.95 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.

6.211.3.96 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

6.211.3.97 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

6.211.3.98 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

6.211.3.99 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.

6.211.3.100 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.

6.211.3.101 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.

6.211.3.102 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

6.211.3.103 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

6.211.3.104 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.
----------------------------------	---

6.211.3.105 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

6.211.3.106 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

Parameters

<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

6.211.3.107 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

6.211.3.108 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

6.211.3.109 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

6.211.3.110 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 <code>SimulationBehaviour</code>
----------	--

Parameters

<i>prefab</i>	<code>SimulationBehaviour</code> used to spawn the <code>NetworkObject</code>
---------------	---

Returns

`T` reference, or null if it was not able to spawn the object

Parameters

<i>obj</i>	Spawned <code>NetworkObject</code> reference
<i>position</i>	Spawn Position
<i>rotation</i>	Spawn Rotation
<i>inputAuthority</i>	Player Input Authority
<i>onBeforeSpawned</i>	<code>OnBeforeSpawned</code> reference
<i>flags</i>	Spawn flags

Type Constraints

T : `SimulationBehaviour`

6.211.3.111 UnloadScene()

```
NetworkSceneAsyncOp UnloadScene (
    string sceneName )
```

Unloads a scene

Parameters

<code>sceneName</code>	Name of the scene to unload
------------------------	-----------------------------

Returns

Scene Unload operation

6.211.3.112 UpdateInternal()

```
void UpdateInternal (
    double dt )
```

This method is meant to be called by [INetworkRunnerUpdater](#).

6.211.4 Property Documentation**6.211.4.1 ActivePlayers**

```
IEnumerable<PlayerRef> ActivePlayers [get]
```

Returns the collection of [PlayerRef](#) objects for this [NetworkRunner](#)'s [Fusion.Simulation](#).

6.211.4.2 AuthenticationValues

```
AuthenticationValues AuthenticationValues [get]
```

[AuthenticationValues](#) used by this Runner to Authenticate the local peer.

6.211.4.3 BuildType

```
BuildTypes BuildType [static], [get]
```

Get [Fusion.Runtime.dll](#) build type.

6.211.4.4 CanSpawn

```
bool CanSpawn [get]
```

Signal if the Network Runner can spawn a [NetworkObject](#)

6.211.4.5 Config

```
NetworkProjectConfig Config [get]
```

Returns the [NetworkProjectConfig](#) reference.

6.211.4.6 CurrentConnectionType

```
ConnectionType CurrentConnectionType [get]
```

Check the current Connection Type with the Remote Server

6.211.4.7 DeltaTime

```
float DeltaTime [get]
```

Returns the fixed tick time interval. Derived from the [SimulationRuntimeConfig.TickRate](#).

6.211.4.8 GameMode

```
GameMode GameMode [get]
```

Current Game Mode active on the [Fusion Simulation](#)

6.211.4.9 Instances

```
IReadOnlyList<NetworkRunner> Instances [static], [get]
```

A list of all [NetworkRunners](#).

6.211.4.10 IsClient

```
bool IsClient [get]
```

Returns if this [Fusion.Simulation](#) represents a Client connection.

6.211.4.11 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

6.211.4.12 IsConnectedToServer

```
bool IsConnectedToServer [get]
```

Returns if this Client is currently connected to a Remote Server

6.211.4.13 IsFirstTick

```
bool IsFirstTick [get]
```

If this is the first tick that executes this update or re-simulation

6.211.4.14 IsForward

```
bool IsForward [get]
```

If this is not a re-simulation but a new forward tick

6.211.4.15 IsLastTick

```
bool IsLastTick [get]
```

If this is the last tick that is being executed this update

6.211.4.16 IsPlayer

```
bool IsPlayer [get]
```

Returns true if this runner represents a Client or Host. Dedicated servers have no local player and will return false.

6.211.4.17 IsResimulation

```
bool IsResimulation [get]
```

If we are currently executing a client side prediction re-simulation.

6.211.4.18 IsResume

```
bool IsResume [get]
```

If this instance is a resume (host migration)

6.211.4.19 IsRunning

```
bool IsRunning [get]
```

Returns if this [Fusion.Simulation](#) is valid and running.

6.211.4.20 IsSceneAuthority

```
bool IsSceneAuthority [get]
```

Is this runner responsible for scene management.

6.211.4.21 IsSceneManagerBusy

```
bool? IsSceneManagerBusy [get]
```

Signals if the [INetworkSceneManager](#) instance assigned to this [NetworkRunner](#) is busy with any scene loading operation.

6.211.4.22 IsServer

```
bool IsServer [get]
```

Returns if this [Fusion.Simulation](#) represents a Server connection.

6.211.4.23 IsSharedModeMasterClient

```
bool IsSharedModeMasterClient [get]
```

Signal if the Local Peer is in a Room and is the Room Master Client

6.211.4.24 IsShutdown

```
bool IsShutdown [get]
```

If the runner is shutdown

6.211.4.25 IsSinglePlayer

```
bool IsSinglePlayer [get]
```

Returns true if this runner was started as single player (Started as [SimulationModes.Host](#) with [SimulationConfig.PlayerCount = 1](#)).

6.211.4.26 IsStarting

```
bool IsStarting [get]
```

If the runner is pending to start

6.211.4.27 LagCompensation

```
HitboxManager LagCompensation [get]
```

Returns the global instance of a lag compensation buffer [Fusion.HitboxManager](#).

6.211.4.28 LatestServerTick

```
Tick LatestServerTick [get]
```

Get the latest confirmed tick of the server we are aware of

6.211.4.29 LobbyInfo

```
LobbyInfo LobbyInfo = new LobbyInfo() [get]
```

Signal if the local peer is already inside a Lobby

6.211.4.30 LocalAlpha

```
float LocalAlpha [get]
```

Get the local time alpha value

6.211.4.31 LocalPlayer

```
PlayerRef LocalPlayer [get]
```

Returns a [PlayerRef](#) for the local simulation. For a dedicated server [PlayerRef.IsRealPlayer](#) will equal false. 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.

6.211.4.32 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).

6.211.4.33 Mode

```
SimulationModes Mode [get]
```

Returns the [SimulationModes](#) flags for The type of network peer the associated [Fusion.Simulation](#) represents.

6.211.4.34 NATType

```
NATType NATType [get]
```

Exposesthe current NAT Type from the local Peer

6.211.4.35 ObjectProvider

```
INetworkObjectProvider ObjectProvider [get]
```

Returns the [INetworkObjectProvider](#) instance.

6.211.4.36 Prefabs

```
NetworkPrefabTable Prefabs [get]
```

Reference to the [NetworkPrefabTable](#).

6.211.4.37 ProvideInput

```
bool ProvideInput [get], [set]
```

Indicates if this [NetworkRunner](#) is collecting [PlayerRef INetworkInput](#).

6.211.4.38 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).

6.211.4.39 SceneManager

```
INetworkSceneManager SceneManager [get]
```

Returns the [INetworkSceneManager](#) instance.

6.211.4.40 SessionInfo

```
SessionInfo SessionInfo = new SessionInfo() [get]
```

Stores information about the current running session

6.211.4.41 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).

6.211.4.42 SimulationUnityScene

```
Scene??? SimulationUnityScene [get]
```

The main scene of the [NetworkRunner](#) or default if not running.

6.211.4.43 Stage

```
SimulationStages Stage [get]
```

Returns the current [SimulationStages](#) stage of this [Fusion.Simulation](#).

6.211.4.44 State

```
States State [get]
```

The current state of the runner, if it's Starting, Running, Shutdown

6.211.4.45 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).

6.211.4.46 TickRate

```
int TickRate [get]
```

6.211.4.47 TicksExecuted

```
int TicksExecuted [get]
```

Returns how many ticks we executed last update.

6.211.4.48 Topology

```
Topologies Topology [get]
```

The current topology used

6.211.4.49 UserId

```
string UserId [get]
```

Photon Client UserID

Returns null if Peer is not connected to Photon Cloud

6.211.5 Event Documentation

6.211.5.1 ObjectAcquired

```
ObjectDelegate ObjectAcquired
```

Event for object acquired

6.212 NetworkRunnerCallbackArgs Class Reference

Stores data types used on the [INetworkRunnerCallbacks](#) interface

Classes

- class [ConnectRequest](#)
Data holder of a Connection Request from a remote client

6.212.1 Detailed Description

Stores data types used on the [INetworkRunnerCallbacks](#) interface

6.213 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

6.213.1 Detailed Description

Data holder of a Connection Request from a remote client

6.213.2 Member Function Documentation

6.213.2.1 Accept()

```
void Accept ( )
```

Accepts the Request

6.213.2.2 Refuse()

```
void Refuse ( )
```

Refuses the Request

6.213.2.3 Waiting()

```
void Waiting ( )
```

Refuses the Request

6.213.3 Property Documentation

6.213.3.1 RemoteAddress

`NetAddress` `RemoteAddress` [get], [set]

Address of the remote client

6.214 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, `NetworkRunnerUpdaterDefaultRenderSettings`)
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

6.214.1 Detailed Description

Default implementation of [INetworkRunnerUpdater](#) that uses the Unity PlayerLoop.

6.214.2 Member Function Documentation

6.214.2.1 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.

6.214.2.2 UnregisterFromPlayerLoop()

```
static bool UnregisterFromPlayerLoop ( ) [static]
```

Unregisters from the PlayerLoop.

Returns

True if unregistered, false if not registered.

6.214.3 Member Data Documentation

6.214.3.1 RenderSettings

```
NetworkRunnerUpdaterDefaultInvokeSettings RenderSettings
```

Initial value:

```
= new NetworkRunnerUpdaterDefaultInvokeSettings {
    ReferencePlayerLoopSystem = typeof(Update.ScriptRunBehaviourUpdate),
    AddMode                  = UnityPlayerLoopSystemAddMode.After
}
```

Default settings for the [NetworkRunner](#) Render Loop.

6.214.3.2 UpdateSettings

```
NetworkRunnerUpdaterDefaultInvokeSettings UpdateSettings
```

Initial value:

```
= new NetworkRunnerUpdaterDefaultInvokeSettings {
    ReferencePlayerLoopSystem = typeof(Update.ScriptRunBehaviourUpdate),
    AddMode                  = UnityPlayerLoopSystemAddMode.Before
}
```

Default settings for the [NetworkRunner](#) Update Loop.

6.215 NetworkRunnerUpdaterDefault.NetworkRunnerRender Struct Reference

Used to invoke [NetworkRunner.RenderInternal](#) in the PlayerLoop.

6.215.1 Detailed Description

Used to invoke [NetworkRunner.RenderInternal](#) in the PlayerLoop.

6.216 NetworkRunnerUpdaterDefault.NetworkRunnerUpdate Struct Reference

Used to invoke [NetworkRunner.UpdateInternal\(double\)](#) in the PlayerLoop.

6.216.1 Detailed Description

Used to invoke [NetworkRunner.UpdateInternal\(double\)](#) in the PlayerLoop.

6.217 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.

6.217.1 Detailed Description

Settings for the [NetworkRunnerUpdaterDefault](#).

6.217.2 Member Function Documentation

6.217.2.1 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.

6.217.2.2 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.

6.217.2.3 GetHashCode()

```
override int GetHashCode ( )
```

Gets the hash code of the settings.

Returns

Hash code.

6.217.2.4 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.

6.217.2.5 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.

6.217.2.6 ToString()

```
override string ToString ( )
```

Returns a string representation of the settings.

Returns

String representation.

6.217.3 Member Data Documentation

6.217.3.1 AddMode

`UnityPlayerLoopSystemAddMode AddMode`

Add mode for the PlayerLoopSystem.

6.217.3.2 ReferencePlayerLoopSystem

Type `ReferencePlayerLoopSystem`

Reference to the PlayerLoopSystem to add the [NetworkRunner](#) to.

6.218 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

6.218.1 Detailed Description

A wrapper for async scene operations.

6.218.2 Member Function Documentation

6.218.2.1 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
---------------	---------------------------

6.218.2.2 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

Generated by Doxygen

Returns

Returns a [NetworkSceneAsyncOp](#) instance

Exceptions

<code>ArgumentNullException</code>	Thrown if <i>asyncOp</i> is null
------------------------------------	----------------------------------

6.218.2.3 FromCompleted()

```
static NetworkSceneAsyncOp FromCompleted (
    SceneRef sceneRef ) [static]
```

Creates a completed [NetworkSceneAsyncOp](#)

Parameters

<code>sceneRef</code>	Scene reference
-----------------------	-----------------

Returns

Returns a [NetworkSceneAsyncOp](#) instance

6.218.2.4 FromCoroutine()

```
static NetworkSceneAsyncOp FromCoroutine (
    SceneRef sceneRef,
    ICoroutine coroutine ) [static]
```

Creates a [NetworkSceneAsyncOp](#) from a [ICoroutine](#)

Parameters

<code>sceneRef</code>	Scene reference
<code>coroutine</code>	Coroutine reference

Returns

Returns a [NetworkSceneAsyncOp](#) instance

Exceptions

<code>ArgumentNullException</code>	Thrown if <i>coroutine</i> is null
------------------------------------	------------------------------------

6.218.2.5 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
------------------------------	--------------------------------

6.218.2.6 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
------------------------------	-------------------------------

6.218.2.7 GetAwaiter()

```
Awaiter GetAwaiter ()
```

Gets the awaiter for the operation

6.218.3 Member Data Documentation

6.218.3.1 SceneRef

```
readonly SceneRef SceneRef
```

The scene reference of the operation

6.218.4 Property Documentation

6.218.4.1 Error

```
Exception? Error [get]
```

Attached error to the operation

6.218.4.2 IsDone

```
bool IsDone [get]
```

Signals if the operation is done

6.218.4.3 IsValid

```
bool IsValid [get]
```

Signals if the operation is valid

6.219 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

6.219.1 Detailed Description

Awaiter for NetworkSceneAsyncOp

6.219.2 Constructor & Destructor Documentation

6.219.2.1 Awaiter()

```
Awaiter (
    in NetworkSceneAsyncOp op )
```

Creates a new `Awaiter` instance

Parameters

<code>op</code>	The operation to await
-----------------	------------------------

6.219.3 Member Function Documentation

6.219.3.1 GetResult()

```
void GetResult ( )
```

Gets the result of the operation

6.219.3.2 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
---------------------	---------------------------

6.219.4 Property Documentation

6.219.4.1 IsCompleted

```
bool IsCompleted [get]
```

Signals if the operation is completed

6.220 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](#), [LocalPhysicsMode](#) 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
- [FixedSize](#)< [NetworkLoadSceneParameters](#) > [SceneParams](#) [get]
The scenes load parameters list
- [FixedSize](#)< [SceneRef](#) > [Scenes](#) [get]
The scenes list
- int [Version](#) [get]
Version number

6.220.1 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.

6.220.2 Member Function Documentation

6.220.2.1 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

6.220.2.2 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

6.220.2.3 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

6.220.2.4 GetHashCode()

```
override int GetHashCode ( )
```

Get the hash code of the [NetworkSceneInfo](#)

Returns

Hash code of the [NetworkSceneInfo](#)

6.220.2.5 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

6.220.2.6 operator NetworkSceneInfo()

```
static implicit operator NetworkSceneInfo (
    SceneRef sceneRef ) [static]
```

Implicit conversion to [NetworkSceneInfo](#)

Parameters

<code>sceneRef</code>	SceneRef to convert
-----------------------	---------------------

Returns

Returns a [NetworkSceneInfo](#) instance

6.220.2.7 RemoveSceneRef()

```
bool RemoveSceneRef (
    SceneRef sceneRef )
```

Removes a scene from the list

Parameters

<code>sceneRef</code>	Scene to remove
-----------------------	-----------------

Returns

Returns true if the scene was removed

6.220.2.8 ToString()

```
override string ToString ( )
```

String representation of the [NetworkSceneInfo](#)

6.220.3 Member Data Documentation**6.220.3.1 MaxScenes**

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

Max number of scenes that can be stored

6.220.3.2 SIZE

```
const int SIZE = 52 [static]
```

The size of the struct in bytes

6.220.3.3 WORD_COUNT

```
const int WORD_COUNT = 13 [static]
```

The size of the struct in words

6.220.4 Property Documentation

6.220.4.1 SceneCount

```
int SceneCount [get]
```

Total Scene Count

6.220.4.2 SceneParams

```
FixedArray<NetworkLoadSceneParameters> SceneParams [get]
```

The scenes load parameters list

6.220.4.3 Scenes

```
FixedArray<SceneRef> Scenes [get]
```

The scenes list

6.220.4.4 Version

```
int Version [get]
```

Version number

6.221 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

Public Attributes

- readonly byte Value
The value of the id

6.221.1 Detailed Description

A unique identifier for a scene load operation.

6.221.2 Constructor & Destructor Documentation

6.221.2.1 NetworkSceneLoadId()

```
NetworkSceneLoadId (
    byte value )
```

Creates a new NetworkSceneLoadId with the given value

Parameters

value	The value of the id
-------	---------------------

6.221.3 Member Function Documentation

6.221.3.1 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

6.221.3.2 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

6.221.3.3 GetHashCode()

```
override int GetHashCode ( )
```

Returns the hash code of the [NetworkSceneLoadId](#)

6.221.4 Member Data Documentation

6.221.4.1 Value

```
readonly byte Value
```

The value of the id

6.222 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.
- override string [ToString \(\)](#)
String representation of the NetworkSceneObjectId.

Public Attributes

- 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.
- int [SceneLoadId](#)
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.

Properties

- bool [IsValid \[get\]](#)
Signal if the NetworkSceneObjectId is valid.

6.222.1 Detailed Description

A unique identifier for a scene object.

6.222.2 Member Function Documentation

6.222.2.1 Equals() [1/2]

```
bool Equals (
    NetworkSceneObjectId other )
```

Check if two NetworkSceneObjectId are equal.

Parameters

<i>other</i>	Another NetworkSceneObjectId to check for equality
--------------	--

Returns

Returns true if the two [NetworkSceneObjectId](#) are equal

6.222.2.2 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

6.222.2.3 GetHashCode()

```
override int GetHashCode ( )
```

Get the hash code of the [NetworkSceneObjectId](#).

6.222.2.4 ToString()

```
override string ToString ( )
```

String representation of the [NetworkSceneObjectId](#).

6.222.3 Member Data Documentation

6.222.3.1 ObjectId

```
int ObjectId
```

Index of the object in the scene or any other form of unique identifier.

6.222.3.2 Scene

```
SceneRef Scene
```

Identifies the scene in which the object is located.

6.222.3.3 SceneLoadId

```
int SceneLoadId
```

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.

6.222.4 Property Documentation

6.222.4.1 IsValid

```
bool IsValid [get]
```

Signal if the [NetworkSceneObjectId](#) is valid.

6.223 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.*

6.223.1 Detailed Description

Network Serialize Method Attribute

6.223.2 Property Documentation

6.223.2.1 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.

6.224 NetworkSimulationConfiguration Class Reference

Configuration for network conditions simulation (induced latency and loss).

Public Member Functions

- [NetworkSimulationConfiguration Clone \(\)](#)
Creates a copy of this NetworkSimulationConfiguration.
- [NetConfigSimulation Create \(\)](#)
Creates a new NetConfigSimulation based on the current configuration.

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](#).

6.224.1 Detailed Description

Configuration for network conditions simulation (induced latency and loss).

6.224.2 Member Function Documentation

6.224.2.1 [Clone\(\)](#)

```
NetworkSimulationConfiguration Clone ( )
```

Creates a copy of this [NetworkSimulationConfiguration](#).

6.224.2.2 [Create\(\)](#)

```
NetConfigSimulation Create ( )
```

Creates a new [NetConfigSimulation](#) based on the current configuration.

Returns

A new [NetConfigSimulation](#) based on the current configuration.

6.224.3 Member Data Documentation

6.224.3.1 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.

6.224.3.2 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.

6.224.3.3 DelayMax

```
double DelayMax = 0.15
```

The highest packet delay value returned from the [DelayShape](#) oscillator.

6.224.3.4 DelayMin

```
double DelayMin = 0.15
```

The lowest packet delay value returned from the [DelayShape](#) oscillator.

6.224.3.5 DelayPeriod

```
double DelayPeriod = 0
```

The period of the [DelayShape](#) oscillator (the rate at which delay oscillates in seconds).

6.224.3.6 DelayShape

```
NetConfigSimulationOscillator.WaveShape DelayShape = NetConfigSimulationOscillator.Wave←  
Shape.Noise
```

The pattern used to oscillate between [DelayMin](#) and [DelayMax](#) values.

6.224.3.7 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](#).

6.224.3.8 Enabled

```
bool Enabled
```

If adverse network conditions are being simulated.

6.224.3.9 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.

6.224.3.10 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.

6.224.3.11 LossChancePeriod

```
double LossChancePeriod = 0
```

The period of the [LossChanceShape](#) oscillator (the rate at which delay oscillates between [LossChanceMin](#) and [LossChanceMax](#)).

6.224.3.12 LossChanceShape

```
NetConfigSimulationOscillator.WaveShape LossChanceShape = NetConfigSimulationOscillator.WaveShape.Noise
```

The pattern used to oscillate between [LossChanceMin](#) and [LossChanceMax](#) values.

6.224.3.13 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](#).

6.225 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

6.225.1 Detailed Description

Spawn Operation

6.225.2 Member Function Documentation

6.225.2.1 GetAwaiter()

```
Awaiter GetAwaiter ()
```

Get this Spawn Operation [Awaiter](#)

6.225.3 Member Data Documentation

6.225.3.1 Runner

```
readonly NetworkRunner Runner
```

Network Runner Reference

6.225.4 Property Documentation

6.225.4.1 IsFailed

```
bool IsFailed [get]
```

Returns true if the object has failed to spawn.

6.225.4.2 IsQueued

```
bool IsQueued [get]
```

Returns true if the object is still queued for spawning.

6.225.4.3 IsSpawned

```
bool IsSpawned [get]
```

Returns true if the object has been spawned.

6.225.4.4 Object

`NetworkObject Object [get]`

Get the spawned Network Object

6.225.4.5 Status

`NetworkSpawnStatus Status [get]`

Get the Spawn Operation Status

6.226 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

6.226.1 Detailed Description

Awaiter for `NetworkSpawnOp`

6.226.2 Constructor & Destructor Documentation

6.226.2.1 Awaiter()

`Awaiter (`
 `in NetworkSpawnOp op)`

`Awaiter` Constructor

Parameters

<i>op</i>	Spawn Operation
-----------	-----------------

6.226.3 Member Function Documentation

6.226.3.1 GetResult()

`NetworkObject GetResult ()`

Get the result of the Spawn Operation

Returns

Spawned Network Object

Exceptions

<code>NetworkObjectSpawnException</code>	Thrown if the Spawn Operation failed
--	--------------------------------------

6.226.3.2 OnCompleted()

`void OnCompleted (`
 Action continuation `)`

`Awaiter` OnCompleted Callback

Parameters

<i>continuation</i>	Continuation Action
---------------------	---------------------

Exceptions

<code>NotSupportedException</code>	Thrown if the Spawn Operation is not supported
------------------------------------	--

6.226.4 Property Documentation

6.226.4.1 IsCompleted

```
bool IsCompleted [get]
```

Returns true if the Spawn Operation is completed

6.227 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.

- **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 ()`
- `IEnumerator IEnumerable. GetEnumerator ()`
- override int `GetHashCode ()`

Returns the hash code for this `NetworkString`.
- 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 character 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 string 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 Unicode code point 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.

6.227.1 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

<code>TSize</code>	
--------------------	--

Type Constraints

- `TSize : unmanaged`**
- `TSize : IFixedStorage`**

6.227.2 Constructor & Destructor Documentation

6.227.2.1 NetworkString()

```
NetworkString (
    string value )
```

Creates a new instance of [NetworkString<Size>](#) with the given value.

Parameters

<code>value</code>	String value.
--------------------	---------------

6.227.3 Member Function Documentation

6.227.3.1 Assign()

```
void Assign (
    string value )
```

Assign a new value to this [NetworkString](#).

Parameters

<code>value</code>	String value.
--------------------	---------------

6.227.3.2 Compare() [1/3]

```
int Compare (
    NetworkString< TSize > s )
```

Compares this instance with a specified [NetworkString](#).

Parameters

<code>s</code>	The NetworkString to compare.
----------------	---

Returns

A 32-bit signed integer that indicates the comparison result.

6.227.3.3 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.

6.227.3.4 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.

6.227.3.5 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*

6.227.3.6 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*

6.227.3.7 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.

6.227.3.8 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.

6.227.3.9 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.

6.227.3.10 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*

6.227.3.11 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*

6.227.3.12 EndsWith()

```
bool EndsWith (
    string s )
```

Checks if the current [NetworkString](#) ends with a specified string.

Parameters

<i>s</i>	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.
------------------------------	---------------------------------

6.227.3.13 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*

6.227.3.14 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.

6.227.3.15 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.

6.227.3.16 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.

6.227.3.17 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.

6.227.3.18 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*

6.227.3.19 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*

6.227.3.20 Get()

```
bool Get (
    ref string cache )
```

Checks if *cache* is equivalent and if not converts to UTF16 and stores the result in *cache*.

Parameters

<i>cache</i>	The string to convert.
--------------	------------------------

Returns

False if no conversion was performed, true otherwise.

6.227.3.21 GetCapacity< TSize >()

```
static int GetCapacity< TSize > ( ) [static]
```

Gets the capacity of a [NetworkString](#) of a specified size.

Template Parameters

<i>TSize</i>	The size of the NetworkString .
--------------	---

Returns

The capacity of a [NetworkString](#) of the specified size.

Type Constraints

TSize : unmanaged

TSize : IFixedStorage

6.227.3.22 GetCharCount()

```
int GetCharCount ( )
```

Calculates the length of the equivalent UTF16 string.

Returns

The length of the equivalent UTF16 string.

6.227.3.23 GetEnumerator()

```
UTF32Tools.CharEnumerator GetEnumerator ( )
```

Returns an enumerator that iterates through the [NetworkString](#).

Returns

A `UTF32Tools.CharEnumerator` for the [NetworkString](#).

6.227.3.24 GetHashCode()

```
override int GetHashCode ( )
```

Returns the hash code for this [NetworkString](#).

Returns

A 32-bit signed integer hash code.

6.227.3.25 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.

6.227.3.26 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.

6.227.3.27 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.

6.227.3.28 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.

6.227.3.29 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.
------------------------------------	---

6.227.3.30 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.

6.227.3.31 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*

6.227.3.32 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*

6.227.3.33 `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>TOtherSize</i> : unmanaged	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](#)**

6.227.3.34 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](#)**

6.227.3.35 operator NetworkString< TSize >()

```
static implicit operator NetworkString< TSize > (
    string str ) [static]
```

Defines an implicit conversion of a string to a [NetworkString](#).

Parameters

<code>str</code>	The string to convert.
------------------	------------------------

Returns

A new instance of [NetworkString](#) with the same value as the string.

6.227.3.36 operator string()

```
static operator string (
    NetworkString< TSize > str ) [explicit], [static]
```

Defines an explicit conversion of a [NetworkString](#) to a string.

Parameters

<code>str</code>	The NetworkString to convert.
------------------	---

Returns

The string value of the [NetworkString](#).

6.227.3.37 operator"!=() [1/3]

```
static bool operator!= (
    NetworkString< TSize > a,
    NetworkString< TSize > b ) [static]
```

Defines an inequality operator for [NetworkString](#).

Parameters

<code>a</code>	The first NetworkString to compare.
<code>b</code>	The second NetworkString to compare.

Returns

true if the NetworkStrings are not equal; otherwise, false.

6.227.3.38 operator"!=() [2/3]

```
static bool operator!= (
    NetworkString< TSize > a,
    string b )  [static]
```

Defines an inequality operator for a [NetworkString](#) and a string.

Parameters

a	The NetworkString to compare.
b	The string to compare.

Returns

true if the [NetworkString](#) and the string are not equal; otherwise, false.

6.227.3.39 operator"!=() [3/3]

```
static bool operator!= (
    string a,
    NetworkString< TSize > b )  [static]
```

Defines an inequality operator for a string and a [NetworkString](#).

Parameters

a	The string to compare.
b	The NetworkString to compare.

Returns

true if the string and the [NetworkString](#) are not equal; otherwise, false.

6.227.3.40 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.

6.227.3.41 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.

6.227.3.42 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.

6.227.3.43 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.

6.227.3.44 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

ArgumentNullException	Thrown when the string is null.
---------------------------------------	---------------------------------

6.227.3.45 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*

6.227.3.46 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.
------------------------------------	--

6.227.3.47 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>	<i>startIndex</i> plus <i>length</i> indicates a position not within this instance, or <i>startIndex</i> or <i>length</i> is less than zero.
------------------------------------	--

6.227.3.48 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.

6.227.3.49 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](#).

6.227.3.50 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.

6.227.4 Property Documentation

6.227.4.1 Capacity

```
int Capacity [get]
```

Maximum UTF32 string length.

6.227.4.2 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.

6.227.4.3 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.

6.227.4.4 Value

```
string Value [get], [set]
```

Converts to/from regular UTF16 string. Setter is alloc-free. Use [Get](#) to get possibly alloc-free conversion.

6.228 NetworkStructUtils Class Reference

Utility methods for [INetworkStruct](#)

Static Public Member Functions

- static int [GetWordCount](#) (Type type)
- static int [GetWordCount< T >](#) ()
Get INetworkStruct Word Count

6.228.1 Detailed Description

Utility methods for [INetworkStruct](#)

6.228.2 Member Function Documentation

6.228.2.1 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](#)

6.229 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

6.229.1 Detailed Description

Describes the total number of WORDs a [INetworkStruct](#) uses.

6.229.2 Constructor & Destructor Documentation

6.229.2.1 NetworkStructWeavedAttribute()

```
NetworkStructWeavedAttribute (
    int wordCount )
```

[NetworkStructWeavedAttribute](#) Constructor

Parameters

<code>wordCount</code>	INetworkStruct word count
------------------------	---

6.229.3 Property Documentation

6.229.3.1 WordCount

```
int WordCount [get]
```

[INetworkStruct](#) Word Count

6.230 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 [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\(\)](#).
- 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

6.230.1 Detailed Description

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

6.230.2 Member Function Documentation

6.230.2.1 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](#).

6.230.2.2 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](#).

6.230.3 Member Data Documentation

6.230.3.1 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\(\)](#).

6.230.3.2 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.

6.230.3.3 SyncScale

```
bool SyncScale = false
```

Enables synchronization of `LocalScale`.

6.230.4 Property Documentation

6.230.4.1 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.

6.231 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 \(NetworkTRSP behaviour, Transform transform, NetworkBehaviourId parentId\)](#)
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.

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

6.231.1 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.

6.231.2 Member Function Documentation

6.231.2.1 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.

6.231.2.2 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

behaviour	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
---------------	--------------------------

6.231.2.3 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](#).

6.231.2.4 SetParentTransform()

```
static void SetParentTransform (
    NetworkTRSP behaviour,
    Transform transform,
    NetworkBehaviourId parentId ) [static], [protected]
```

Default handling for setting a [NetworkTRSP](#)'s parent using a [NetworkBehaviourId](#) value.

6.231.2.5 Teleport()

```
static void Teleport (
    NetworkTRSP behaviour,
    Transform transform,
    Vector3? position = null,
    Quaternion? rotation = null ) [static], [protected]
```

The default Teleport implementation for [NetworkTRSP](#) derived classes.

6.231.3 Property Documentation

6.231.3.1 Data

```
NetworkTRSPData Data [get]
```

The networked data of this [NetworkTRSP](#).

6.231.3.2 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](#).

6.231.3.3 State

```
ref NetworkTRSPData State [get], [protected]
```

A reference to the networked data of this [NetworkTRSP](#).

6.232 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](#)
- [NetworkBehaviour 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

6.232.1 Detailed Description

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

6.232.2 Member Data Documentation

6.232.2.1 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](#)

6.232.2.2 Parent

[NetworkBehaviourId](#) **Parent**

Id of a [NetworkBehaviour](#) on the parent of the component's transform.

6.232.2.3 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)

6.232.2.4 POSITION_OFFSET

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

Offset to point at the position values on the data buffer

6.232.2.5 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)

6.232.2.6 Scale

`Vector3Compressed` Scale

Scale relevant for the spatial synchronization component

6.232.2.7 SIZE

```
const int SIZE = WORDS * Allocator.REPLICATE_WORD_SIZE [static]
```

The actual size for the networked properties in bytes

6.232.2.8 TeleportKey

int TeleportKey

Key used to differentiate between several teleports

6.232.2.9 WORDS

```
const int WORDS = 14 [static]
```

Networked properties word count for the base `NetworkTRSPData`

6.232.3 Property Documentation

6.232.3.1 NonNetworkedParent

`NetworkBehaviourId` NonNetworkedParent [static], [get]

Special `NetworkBehaviourId` value, used as a flag to tell the parent is a non-networked object

6.233 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

6.233.1 Detailed Description

Enables a special inspector drawer for Unity Rect type, specially designed for editing RectTransforms using normalized values.

6.233.2 Constructor & Destructor Documentation

6.233.2.1 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

<code>invertY</code>	Invert Y handling
<code>aspectRatio</code>	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.

6.233.3 Member Data Documentation

6.233.3.1 AspectRatio

`float AspectRatio`

Set the Aspect Ratio

6.233.3.2 InvertY

`bool InvertY`

Signal if Y should be inverted

6.234 OnChangedRenderAttribute Class Reference

OnChangedRender Attribute

Inherits Attribute.

Public Member Functions

- [OnChangedRenderAttribute](#) (string methodName)
Initializes a new instance of the `OnChangedRenderAttribute` class.

Properties

- string [MethodName](#) [get]
Gets the name of the method to be called when the property changes.

6.234.1 Detailed Description

OnChangedRender Attribute

This attribute is used to specify a method that should be called when the property changes.

6.234.2 Constructor & Destructor Documentation

6.234.2.1 OnChangedRenderAttribute()

```
OnChangedRenderAttribute (
    string methodName )
```

Initializes a new instance of the [OnChangedRenderAttribute](#) class.

Parameters

<code>methodName</code>	The name of the method to be called when the property changes.
-------------------------	--

Exceptions

<code>ArgumentNullException</code>	Thrown when <code>methodName</code> is null or empty.
------------------------------------	---

6.234.3 Property Documentation

6.234.3.1 MethodName

```
string MethodName [get]
```

Gets the name of the method to be called when the property changes.

6.235 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.
- 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.

6.235.1 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
```

6.235.2 Member Function Documentation

6.235.2.1 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.

6.235.2.2 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.

6.235.2.3 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.

6.235.2.4 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.

6.235.2.5 GetHashCode()

```
override int GetHashCode ( )
```

Serves as the default hash function.

6.235.2.6 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.

6.235.2.7 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.

6.235.2.8 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.

6.235.2.9 `ToString()`

```
override string ToString ( )
```

Returns a string that represents the current object.

6.235.2.10 `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.

6.235.2.11 `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*

6.235.3 Member Data Documentation

6.235.3.1 MASTER_CLIENT_RAW

```
const int MASTER_CLIENT_RAW = -1 [static]
```

A constant representing the raw index value for the master client.

6.235.3.2 SIZE

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

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

6.235.4 Property Documentation

6.235.4.1 AsIndex

```
int AsIndex [get]
```

Returns the [PlayerRef](#) int as an integer Id value.

-1=None -2=MasterClient >=0=PlayerId

6.235.4.2 Comparer

```
IEqualityComparer<PlayerRef> Comparer = new IndexEqualityComparer() [static], [get]
```

Gets an equality comparer that can be used to compare two [PlayerRef](#) instances.

6.235.4.3 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.State Authority to indicate that the MasterClient has authority (which ever player that currently is), rather than a specific Player.

6.235.4.4 IsNone

```
bool IsNone [get]
```

Returns true if the index value equals -1 (internal raw value of 0), indicating no player.

6.235.4.5 IsRealPlayer

```
bool IsRealPlayer [get]
```

If this player ref is a valid unique player index

6.235.4.6 MasterClient

```
PlayerRef MasterClient [static], [get]
```

Special master client player ref value of -1

6.235.4.7 None

```
PlayerRef None [static], [get]
```

None player

6.235.4.8 PlayerId

```
int PlayerId [get]
```

Returns the [PlayerRef](#) as an integer Id value.

-1=None -2=MasterClient

6.235.4.9 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

6.236 PreserveInPluginAttribute Class Reference

Preserve In Plugin Attribute

Inherits Attribute.

Public Member Functions

- [PreserveInPluginAttribute \(\)](#)
PreserveInPluginAttribute Constructor

6.236.1 Detailed Description

Preserve In Plugin Attribute

6.236.2 Constructor & Destructor Documentation

6.236.2.1 PreserveInPluginAttribute()

`PreserveInPluginAttribute ()`

`PreserveInPluginAttribute` Constructor

6.237 IMessage Interface Reference

Represents a [Protocol](#) Message

6.237.1 Detailed Description

Represents a [Protocol](#) Message

Used to tag the Messages in ICommunicator.

6.238 Versioning Class Reference

[Versioning](#) Information

Static Public Attributes

- static readonly Version **InvalidVersion** = new Version(0, 0, 0)

Properties

- static Version [GetCurrentVersion](#) [get]
Get the current version
- static string [GetProductVersion](#) [get]
Get the current product version

6.238.1 Detailed Description

[Versioning](#) Information

6.238.2 Property Documentation

6.238.2.1 GetCurrentVersion

Version [GetCurrentVersion](#) [static], [get]

Get the current version

6.238.2.2 GetProductVersion

string [GetProductVersion](#) [static], [get]

Get the current product version

6.239 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`

6.239.1 Detailed Description

`Ptr`

6.239.2 Member Function Documentation

6.239.2.1 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

6.239.2.2 Equals() [2/2]

```
bool Equals (
    Ptr other )
```

Check [Ptr](#) equality

Parameters

<i>other</i>	Ptr Ref
--------------	-------------------------

Returns

True if points to the same Address

6.239.2.3 GetHashCode()

```
override int GetHashCode ( )
```

[Ptr](#) Hash Code, same as [Address](#)

Returns

[Address](#)

6.239.2.4 operator bool()

```
static implicit operator bool (
    Ptr a ) [static]
```

Implicit Bool Operator Check if [Address](#) is not 0

Parameters

a	Ptr to check
---	--------------

Returns

True if [Address](#) is not 0

6.239.2.5 operator"!=()

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

Implicit [Ptr](#) Not Equals Operator

Parameters

a	Ptr A
b	Ptr B

Returns

True if [Address](#) is not the same

6.239.2.6 operator+()

```
static Ptr operator+ (
    Ptr p,
    int v ) [static]
```

Implicit [Ptr](#) Sum Operator

Parameters

p	Ptr to add to
v	Value to add

Returns

[Ptr](#) with [Address](#) increased by *v*

6.239.2.7 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*

6.239.2.8 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

6.239.2.9 ToString()

```
override string ToString ( )
```

Ptr to String

Returns

Address in Hexadecimal format

6.239.3 Member Data Documentation

6.239.3.1 Address

int Address

Ptr Address

6.239.3.2 SIZE

const int SIZE = 4 [static]

Ptr Size

6.239.4 Property Documentation

6.239.4.1 Null

Ptr Null [static], [get]

Null Ptr

6.240 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

6.240.1 Detailed Description

Ptr Equality Comparer

6.240.2 Member Function Documentation

6.240.2.1 Equals()

```
bool Equals (
```

`Ptr x,`

`Ptr y)`

`Ptr Equality Comparer`

Parameters

x	Ptr X
y	Ptr Y

Returns

True if point to the same Address

6.240.2.2 GetHashCode()

```
int GetHashCode (
    Ptr obj )
```

Get Hash Code

Parameters

obj	Ptr
-----	-----

Returns

Ptr Address

6.241 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.

6.241.1 Detailed Description

Represents a compressed Quaternion value for network transmission.

6.241.2 Member Function Documentation

6.241.2.1 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

<code>obj</code>	The object to compare with the current <code>QuaternionCompressed</code> instance.
------------------	--

Returns

True if the provided object is a `QuaternionCompressed` instance and it's equal to the current `QuaternionCompressed` instance, otherwise false.

6.241.2.2 Equals() [2/2]

```
bool Equals (
    QuaternionCompressed other )
```

Checks if the current `QuaternionCompressed` instance is equal to the other `QuaternionCompressed` instance.

Parameters

<code>other</code>	The other <code>QuaternionCompressed</code> instance to compare with the current <code>QuaternionCompressed</code> instance.
--------------------	--

Returns

True if the values of both `QuaternionCompressed` instances are equal, otherwise false.

6.241.2.3 GetHashCode()

```
override int GetHashCode ( )
```

Returns the hash code for the current `QuaternionCompressed` instance.

Returns

A hash code for the current `QuaternionCompressed` instance.

6.241.2.4 operator Quaternion()

```
static implicit operator Quaternion (
    QuaternionCompressed q ) [static]
```

Implicit conversion from `QuaternionCompressed` to `Quaternion`.

Parameters

<code>q</code>	The <code>QuaternionCompressed</code> instance to convert.
----------------	--

Returns

The decompressed `Quaternion` value of the `QuaternionCompressed` instance.

6.241.2.5 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.

6.241.2.6 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.

6.241.2.7 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.

6.241.3 Member Data Documentation

6.241.3.1 wEncoded

```
int wEncoded
```

Encoded value of the w component.

6.241.3.2 xEncoded

```
int xEncoded
```

Encoded value of the x component.

6.241.3.3 yEncoded

```
int yEncoded
```

Encoded value of the y component.

6.241.3.4 zEncoded

```
int zEncoded
```

Encoded value of the z component.

6.241.4 Property Documentation

6.241.4.1 W

```
float W [get], [set]
```

Gets or sets the w component.

6.241.4.2 X

```
float X [get], [set]
```

Gets or sets the x component.

6.241.4.3 Y

```
float Y [get], [set]
```

Gets or sets the y component.

6.241.4.4 Z

```
float Z [get], [set]
```

Gets or sets the z component.

6.242 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 [NetworkBehaviour](#) [ReadNetworkBehaviourRef](#) (int *data, [NetworkRunner](#) runner, out bool isValid)
Reads a NetworkBehaviour reference from the provided memory location. Null is considered valid (0,1).
- 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 [WriteEmptyNetworkBehaviourRef](#) (int *data)
Writes an empty NetworkBehaviour reference to the provided memory location.
- static void [WriteFloat](#) (int *data, float f)
Writes a float value to the provided memory location.
- static void [WriteNetworkBehaviourRef](#) (int *data, [NetworkRunner](#) runner, [NetworkBehaviour](#) reference)
Writes a NetworkBehaviour reference to the provided memory location.
- static void [WriteNullBehaviourRef](#) (int *data)
Writes a null NetworkBehaviour reference 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.

6.242.1 Detailed Description

Provides utility methods for reading and writing data.

6.242.2 Member Function Documentation

6.242.2.1 [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.

6.242.2.2 ReadNetworkBehaviourRef()

```
static NetworkBehaviour ReadNetworkBehaviourRef (
    int * data,
    NetworkRunner runner,
    out bool isValid ) [static]
```

Reads a [NetworkBehaviour](#) reference from the provided memory location. Null is considered valid (0,1).

Parameters

<i>data</i>	The memory location to read from.
<i>runner</i>	The NetworkRunner associated with the NetworkBehaviour .
<i>isValid</i>	Out parameter indicating whether the read operation was valid.

Returns

The [NetworkBehaviour](#) reference read from the memory location.

6.242.2.3 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.

6.242.2.4 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.

6.242.2.5 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.

6.242.2.6 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.

6.242.2.7 WriteEmptyNetworkBehaviourRef()

```
static void WriteEmptyNetworkBehaviourRef (
    int * data ) [static]
```

Writes an empty [NetworkBehaviour](#) reference to the provided memory location.

Parameters

<i>data</i>	The memory location to write to.
-------------	----------------------------------

6.242.2.8 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.

6.242.2.9 WriteNetworkBehaviourRef()

```
static void WriteNetworkBehaviourRef (
    int * data,
    NetworkRunner runner,
    NetworkBehaviour reference ) [static]
```

Writes a [NetworkBehaviour](#) reference to the provided memory location.

Parameters

<i>data</i>	The memory location to write to.
<i>runner</i>	The NetworkRunner associated with the NetworkBehaviour .
<i>reference</i>	The NetworkBehaviour reference to write.

6.242.2.10 WriteNullBehaviourRef()

```
static void WriteNullBehaviourRef (
    int * data ) [static]
```

Writes a null [NetworkBehaviour](#) reference to the provided memory location.

Parameters

<i>data</i>	The memory location to write to.
-------------	----------------------------------

6.242.2.11 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.

6.242.2.12 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.

6.242.2.13 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.

6.242.2.14 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.

6.242.3 Member Data Documentation**6.242.3.1 ACCURACY**

```
const float ACCURACY = 1 << 10 [static]
```

Accuracy of floating point values when serialized.

6.243 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)

- static unsafe int [ReadStringUtf32WithHash](#) (int *ptr, int maxLength, ref string cache)

Reads a string from the provided memory location in UTF32 format without a hash.
- static int [ReadStringUtf8NoHash](#) (void *source, out string result)

Reads a string from the provided memory location in UTF8 format with 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.

6.243.1 Detailed Description

Provides utility methods for reading and writing data.

6.243.2 Member Function Documentation

6.243.2.1 GetByteArrayHashCode()

```
static unsafe int GetByteArrayHashCode (
    byte * ptr,
    int length ) [static]
```

Parameters

<i>ptr</i>	
<i>length</i>	

Returns

6.243.2.2 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.

6.243.2.3 GetStringHashCode()

```
static int GetStringHashCode (
    string value,
    int maxLength ) [static]
```

Parameters

<i>value</i>	
<i>maxLength</i>	

Returns

6.243.2.4 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.

6.243.2.5 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.

6.243.2.6 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.

6.243.2.7 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.

6.243.2.8 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.

6.243.2.9 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.

6.243.2.10 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.

6.243.2.11 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.

6.243.2.12 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.

6.243.2.13 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.

6.243.2.14 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.

6.244 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.

6.244.1 Detailed Description

Provides utility methods for reflection.

6.244.2 Member Function Documentation

6.244.2.1 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](#).

6.244.2.2 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](#).

6.244.2.3 GetAllWeavedAssemblies()

```
static IEnumerable<Assembly> GetAllWeavedAssemblies () [static]
```

Gets all assemblies that have been weaved.

Returns

An `IEnumerable` of all weaved assemblies.

6.244.2.4 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.

6.244.2.5 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.

6.244.2.6 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.

6.244.2.7 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

<code>T</code>	The type of the attribute to retrieve. Must be a subclass of <code>Attribute</code> .
----------------	---

Parameters

<code>member</code>	The member to retrieve the attribute from.
<code>inherit</code>	Specifies whether to search this member's inheritance chain to find the attributes.

Returns

The custom attribute of type `T`.

Exceptions

<code>ArgumentOutOfRangeException</code>	Thrown when the provided member does not have an attribute of type <code>T</code> .
<code>InvalidOperationException</code>	Thrown when the provided member has more than one attribute of type <code>T</code> .

Type Constraints

T : Attribute

6.244.2.8 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.
----------------------------------	---

6.245 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]

6.245.1 Detailed Description

Override default render settings for [Networked] properties.

6.245.2 Constructor & Destructor Documentation

6.245.2.1 RenderAttribute() [1/2]

```
RenderAttribute ()
```

Default constructor for [RenderAttribute](#)

6.245.2.2 RenderAttribute() [2/2]

```
RenderAttribute (
    RenderTimeframe timeframe,
    RenderSource source )
```

[RenderAttribute](#) Constructor

Parameters

<i>timeframe</i>	RenderTimeframe reference
<i>source</i>	RenderSource reference

6.245.3 Property Documentation

6.245.3.1 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) { /* ... */ }
```

6.245.3.2 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.

6.245.3.3 Timeframe

```
RenderTimeframe Timeframe [get], [set]
```

Force this property to be rendered in this [RenderTimeframe](#).

This setting is prioritized over [NetworkBehaviour](#) and [NetworkObject](#) overrides.

6.246 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](#).

6.246.1 Detailed Description

Can be used to acquire interpolated data for different points in time.

6.246.2 Member Function Documentation

6.246.2.1 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.

6.247 RenderWeavedAttribute Class Reference

Render Weaved Attribute

Inherits Attribute.

Public Member Functions

- [RenderWeavedAttribute \(\)](#)
RenderWeavedAttribute Constructor

6.247.1 Detailed Description

Render Weaved Attribute

6.247.2 Constructor & Destructor Documentation

6.247.2.1 RenderWeavedAttribute()

`RenderWeavedAttribute ()`

RenderWeavedAttribute Constructor

6.248 ResolveNetworkPrefabSourceAttribute Class Reference

Resolve Network Prefab Source Attribute

Inherits PropertyAttribute.

6.248.1 Detailed Description

Resolve Network Prefab Source Attribute

6.249 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.SourcesServer \[get, set\]](#)
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.

6.249.1 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 [RpclInfo](#) 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, RpclInfo  
info = default) {} To target a specific Player, use the RpcTargetAttribute: | [Rpc] | public  
void RpcFoo([RpcTarget] PlayerRef targetPlayer) {} Use RpclInfo 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) {}

6.249.2 Constructor & Destructor Documentation

6.249.2.1 RpcAttribute() [1/2]

```
RpcAttribute ( )
```

Constructor for RpcAttributes.

6.249.2.2 RpcAttribute() [2/2]

```
RpcAttribute (
    RpcSources sources,
    RpcTargets targets )
```

Constructor for RpcAttributes.

Parameters

<code>sources</code>	The legal RpcSources types that can trigger this Rpc. Default is RpcSources.All
<code>targets</code>	The RpcTargets types that will receive and invoke this method. Default is RpcTargets.All

6.249.3 Member Data Documentation

6.249.3.1 MaxPayloadSize

```
const int MaxPayloadSize = SimulationMessage.MAX\_PAYLOAD\_SIZE [static]
```

Maximum allowed size for the payload of the RPC message.

6.249.4 Property Documentation

6.249.4.1 Channel

```
RpcChannel Channel = RpcChannel.Reliable [get], [set]
```

Specifies which RpcChannel to use. Default value is [RpcChannel.Reliable](#)

6.249.4.2 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.

6.249.4.3 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.

6.249.4.4 Sources

```
int Sources [get]
```

The legal [RpcSources](#) types that can trigger this Rpc. Cast to int. Default value is (int)[RpcSources.All](#).

6.249.4.5 Targets

```
int Targets [get]
```

The [RpcTargets](#) types that will receive and invoke this method. Cast to int. Default value is (int)[RpcTargets.All](#).

6.249.4.6 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.

6.250 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.

6.250.1 Detailed Description

Header for RPC messages.

6.250.2 Member Function Documentation

6.250.2.1 [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.

6.250.2.2 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.

6.250.2.3 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.

6.250.2.4 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.

6.250.2.5 ToString()

```
override string ToString ( )
```

Returns a string that represents the current [RpcHeader](#).

Returns

Returns a string that represents the current [RpcHeader](#).

6.250.2.6 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.

6.250.3 Member Data Documentation

6.250.3.1 Behaviour

ushort [Behaviour](#)

The behaviour associated with the RPC message.

6.250.3.2 Method

ushort [Method](#)

The method associated with the RPC message.

6.250.3.3 Object

[NetworkId](#) [Object](#)

The [NetworkId](#) of the object associated with the RPC message.

6.250.3.4 SIZE

const int SIZE = NetworkId.SIZE + 2 + 2 [static]

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

6.251 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.

6.251.1 Detailed Description

[RpcInfo](#) is a struct that contains information about the RPC message.

6.251.2 Member Function Documentation

6.251.2.1 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.

6.251.2.2 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.

6.251.2.3 ToString()

```
override string ToString ()
```

Returns a string that represents the current [RpcInfo](#).

Returns

Returns a string that represents the current [RpcInfo](#).

6.251.3 Member Data Documentation

6.251.3.1 Channel

```
RpcChannel Channel
```

Represents the channel through which the RPC message was sent.

6.251.3.2 IsInvokeLocal

```
bool IsInvokeLocal
```

Indicates whether the RPC message is invoked locally.

6.251.3.3 Source

```
PlayerRef Source
```

Represents the player who sent the RPC message.

6.251.3.4 Tick

`Tick` `Tick`

Represents the tick at which the RPC message was sent.

6.252 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.

6.252.1 Detailed Description

Represents the data required to invoke an RPC message.

6.252.2 Member Function Documentation

6.252.2.1 `ToString()`

`override string ToString ()`

Returns a string that represents the current `RpcInvokeData`.

Returns

Returns a string that represents the current `RpcInvokeData`.

6.252.3 Member Data Documentation

6.252.3.1 Delegate

`RpcInvokeDelegate` Delegate

Represents the delegate to be invoked for the RPC message.

6.252.3.2 Key

`int Key`

Represents the key associated with the RPC message.

6.252.3.3 Sources

`int Sources`

Represents the sources of the RPC message.

6.252.3.4 Targets

`int Targets`

Represents the targets of the RPC message.

6.253 `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.

6.253.1 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); | }
```

6.253.2 Member Function Documentation

6.253.2.1 ToString()

```
override string ToString ( )
```

Returns a string that represents the current [RpclnvokelInfo](#).

6.253.3 Member Data Documentation

6.253.3.1 LocalInvokeResult

```
RpcLocalInvokeResult LocalInvokeResult
```

Represents the result of the local RPC invocation.

6.253.3.2 SendCullResult

```
RpcSendCullResult SendCullResult
```

Represents the result of the RPC message send operation.

6.253.3.3 SendResult

[RpcSendResult](#) SendResult

Contains detailed information about the RPC send operation result.

6.254 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.

6.254.1 Detailed Description

RPC send operation result information.

6.254.2 Member Function Documentation

6.254.2.1 [ToString\(\)](#)

override string [ToString](#) ()

Returns a string that represents the current [RpcSendResult](#).

6.254.3 Member Data Documentation

6.254.3.1 MessageSize

```
int MessageSize
```

The size of the RPC message.

6.254.3.2 Result

```
RpcSendMessageResult Result
```

Result flags for the RPC send operation.

6.255 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.

6.255.1 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) { }
```

6.255.2 Constructor & Destructor Documentation

6.255.2.1 RpcTargetAttribute()

```
RpcTargetAttribute ( )
```

RPC Attribute constructor.

6.256 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.

6.256.1 Detailed Description

Struct that contains information about a scene after it has been loaded.

6.256.2 Constructor & Destructor Documentation

6.256.2.1 SceneLoadDoneArgs()

```
SceneLoadDoneArgs (
    SceneRef sceneRef,
    NetworkObject[] sceneObjects,
    Scene scene = default,
    GameObject[] rootGameObjects = default )
```

Constructs a new `SceneLoadDoneArgs` struct.

Parameters

<code>sceneRef</code>	Reference to the loaded scene.
<code>sceneObjects</code>	Array of NetworkObjects present in the loaded scene.
<code>scene</code>	The loaded Unity scene.
<code>rootGameObjects</code>	Array of root GameObjects present in the loaded Unity scene.

6.256.3 Member Data Documentation

6.256.3.1 RootGameObjects

```
readonly GameObject [ ] RootGameObjects
```

Array of root GameObjects present in the loaded Unity scene.

6.256.3.2 Scene

```
readonly Scene Scene
```

The loaded Unity scene.

6.256.3.3 SceneObjects

```
readonly NetworkObject [ ] SceneObjects
```

Array of NetworkObjects present in the loaded scene.

6.256.3.4 SceneRef

```
readonly SceneRef SceneRef
```

Reference to the loaded scene.

6.257 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 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.

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

6.257.1 Detailed Description

Scene reference struct. Can be used to reference a scene by index or by path.

6.257.2 Member Function Documentation

6.257.2.1 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.

6.257.2.2 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.

6.257.2.3 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

ArgumentOutOfRangeException	Thrown when the index is less than 0 or equal to int.MaxValue.
---	--

6.257.2.4 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 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.

6.257.2.5 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.

6.257.2.6 GetHashCode()

```
override int GetHashCode ( )
```

Serves as the default hash function.

Returns

A hash code for the current [SceneRef](#).

6.257.2.7 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.

6.257.2.8 operator"!=()

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

Returns true if the values are not equal.

Parameters

a	SceneRef a
b	SceneRef b

Returns

true if the values are not equal; otherwise, false.

6.257.2.9 operator==()

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

Returns true if the values are equal.

Parameters

a	SceneRef a
b	SceneRef b

Returns

true if the values are equal; otherwise, false.

6.257.2.10 ToString() [1/2]

```
override string ToString ( )
```

Returns a string that represents the current [SceneRef](#).

Returns

A string that represents the current [SceneRef](#).

6.257.2.11 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.

6.257.3 Member Data Documentation**6.257.3.1 FLAG_ADDRESSABLE**

```
const uint FLAG_ADDRESSABLE = 1u << 31 [static]
```

A constant representing the flag for addressable scenes.

6.257.3.2 RawValue

```
uint RawValue
```

The raw value of the [SceneRef](#). This can represent either an index or a path hash, depending on the flag.

6.257.3.3 SIZE

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

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

6.257.4 Property Documentation

6.257.4.1 AsIndex

```
int AsIndex [get]
```

Returns lower 32 bits as an index.

6.257.4.2 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.
----------------------------------	---

6.257.4.3 IsIndex

```
bool IsIndex [get]
```

Returns true if this scene ref is an index.

6.257.4.4 IsValid

```
bool IsValid [get]
```

If this scene index is valid

6.257.4.5 None

```
SceneRef None [static], [get]
```

None scene

6.258 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 `EntryKeyPropertyName` = nameof(Entry.Key)
The property path for the key in the Entry structure.
- const string `ItemsPropertyName` = 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.

6.258.1 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.

6.258.2 Member Function Documentation

6.258.2.1 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.

6.258.2.2 Clear()

```
virtual void Clear ( ) [virtual]
```

Removes all keys and values from the [SerializableDictionary](#).

6.258.2.3 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.

6.258.2.4 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.

6.258.2.5 GetEnumerator()

```
Dictionary<TKey, TValue>.Enumerator GetEnumerator ( )
```

Returns an enumerator that iterates through the [SerializableDictionary](#).

Returns

A Dictionary{*TKey*,*TValue*}.Enumerator structure for the [SerializableDictionary](#).

6.258.2.6 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](#).

6.258.2.7 Reset()

```
void Reset ( )
```

Resets the [SerializableDictionary](#), clearing its internal dictionary.

6.258.2.8 Store()

```
void Store ( )
```

Stores the [SerializableDictionary](#)'s data into an array for serialization. This includes handling duplicates and null keys.

6.258.2.9 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.

6.258.2.10 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.

6.258.3 Member Data Documentation

6.258.3.1 EntryKeyPropertyName

```
const string EntryKeyPropertyName = nameof(Entry.Key) [static]
```

The property path for the key in the Entry structure.

6.258.3.2 ItemsPropertyName

```
const string ItemsPropertyName = nameof(_items) [static]
```

The property path for the items in the [SerializableDictionary](#).

6.258.4 Property Documentation

6.258.4.1 Count

```
int Count [get]
```

Gets the number of key/value pairs contained in the [SerializableDictionary](#).

6.258.4.2 IsReadOnly

```
bool IsReadOnly [get]
```

Gets a value indicating whether the [SerializableDictionary](#) is read-only. This value is always false.

6.258.4.3 Keys

```
Dictionary< TKey, TValue>.KeyCollection Keys [get]
```

Gets a collection containing the keys in the [SerializableDictionary](#).

6.258.4.4 this[TKey key]

```
TValue this[TKey key] [get], [set]
```

Gets or sets the value associated with the specified key.

Parameters

<code>key</code>	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.

6.258.4.5 Values

`Dictionary< TKey, TValue>.ValueCollection Values [get]`

Gets a collection containing the values in the [SerializableDictionary](#).

6.259 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

6.259.1 Detailed Description

Holds information about the Game Session

6.259.2 Member Function Documentation

6.259.2.1 operator bool()

```
static implicit operator bool (
    SessionInfo sessionInfo ) [static]
```

Check if the [SessionInfo](#) reference is not Null and is Valid.

Parameters

<i>sessionInfo</i>	Session Info
--------------------	--------------

6.259.2.2 ToString()

```
override string ToString ( )
```

String representation of a [SessionInfo](#)

Returns

Formatted [SessionInfo](#)

6.259.2.3 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
-------------------------	-----------------------

6.259.3 Property Documentation

6.259.3.1 IsOpen

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

Signal if the current connected Room is open

6.259.3.2 IsValid

```
bool IsValid [get]
```

Flag to signal if the [SessionInfo](#) is ready for use

6.259.3.3 IsVisible

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

Signal if the current connected Room is visible

6.259.3.4 MaxPlayers

```
int MaxPlayers [get]
```

Max number of peer that can join this Session, this value always include an extra slot for the Server/Host

6.259.3.5 Name

```
string Name [get]
```

Stores the current Room Name

6.259.3.6 PlayerCount

```
int PlayerCount [get]
```

Current number of peers inside this Session, this includes the Server/Host and Clients

6.259.3.7 Properties

```
ReadOnlyDictionary<string, SessionProperty> Properties [get]
```

Room Custom Properties

6.259.3.8 Region

```
string Region [get]
```

Stores the current connected Region

6.260 Simulation Class Reference

Main simulation class

Inherits ILogSourceProxy, 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.
- 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)

- 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 StateAuthority
- 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]

- 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.

6.260.1 Detailed Description

Main simulation class

6.260.2 Member Function Documentation

6.260.2.1 AfterSimulation()

```
virtual void AfterSimulation ( ) [protected], [virtual]
```

Callback invoked after the [Simulation](#) Update

6.260.2.2 AfterUpdate()

```
virtual void AfterUpdate ( ) [protected], [virtual]
```

Callback invoked After the [Simulation](#) Update

6.260.2.3 BeforeFirstTick()

```
virtual void BeforeFirstTick ( ) [protected], [virtual]
```

Callback invoked before the First [Tick](#)

6.260.2.4 BeforeSimulation()

```
virtual int BeforeSimulation ( ) [protected], [virtual]
```

Callback invoked before the [Simulation](#) Loop

Returns

Total number of re-simulations

6.260.2.5 BeforeUpdate()

```
virtual void BeforeUpdate ( ) [protected], [virtual]
```

Callback invoked before the [Simulation](#) Update

6.260.2.6 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.
---------------	--

6.260.2.7 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](#)

6.260.2.8 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

6.260.2.9 GetObjectsAndPlayersInAreaOfInterestCell()

```
void GetObjectsAndPlayersInAreaOfInterestCell (
    int cellKey,
    List< PlayerRef > players,
    List< NetworkId > objects )
```

Used by RunnerAOIGizmos component. Supplies data about current active AOI cells.

6.260.2.10 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](#)

6.260.2.11 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.

6.260.2.12 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

6.260.2.13 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

6.260.2.14 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

6.260.2.15 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

6.260.2.16 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

6.260.2.17 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

6.260.2.18 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

6.260.2.19 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

6.260.2.20 NetworkConnected()

```
virtual void NetworkConnected (
    NetConnection * connection ) [protected], [virtual]
```

Callback invoked on the Connected

Parameters

<i>connection</i>	Connection that was connected
-------------------	-------------------------------

6.260.2.21 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

6.260.2.22 NetworkReceiveDone()

```
virtual void NetworkReceiveDone ( ) [protected], [virtual]
```

Callback invoked when the Network Receive is completed

6.260.2.23 NoSimulation()

```
virtual void NoSimulation ( ) [protected], [virtual]
```

Callback invoked when there is no simulation

6.260.2.24 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

6.260.2.25 Update()

```
int Update (
    double dt )
```

Forwards the [Simulation](#) based on the Delta Time

Parameters

<code>dt</code>	Delta Time used to forward the simulation
-----------------	---

Returns

How many Ticks executed on this Update

6.260.3 Property Documentation

6.260.3.1 ActivePlayers

```
virtual IEnumerable<PlayerRef> ActivePlayers [get]
```

List of Active players in the [Simulation](#).

6.260.3.2 Config

```
SimulationConfig Config [get]
```

The [SimulationConfig](#) file used by this [Simulation](#).

6.260.3.3 DeltaTime

```
float DeltaTime [get]
```

Gets the fixed tick time interval. Derived from the [SimulationRuntimeConfig.TickRate](#).

6.260.3.4 InputCount

```
int InputCount [get]
```

The current input collection size

6.260.3.5 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.

6.260.3.6 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.

6.260.3.7 IsForward

```
bool IsForward [get]
```

Use inside of FixedUpdateNetwork to determine if the tick currently being simulated has NOT previously been simulated locally.

6.260.3.8 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.

6.260.3.9 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.

6.260.3.10 IsMasterClient

```
bool IsMasterClient [get]
```

Only valid in Shared Mode. Indicates if this peer is flagged as the MasterClient, which means it is default StateAuthority

6.260.3.11 IsPlayer

```
bool IsPlayer [get]
```

True for any peer that represents a human player. This is true for all peers except a dedicated server.

6.260.3.12 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.

6.260.3.13 IsRunning

```
bool IsRunning [get]
```

Signal if the [Simulation](#) is currently running

6.260.3.14 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).

6.260.3.15 IsShutdown

```
bool IsShutdown [get]
```

6.260.3.16 IsSinglePlayer

```
bool IsSinglePlayer [get]
```

Indicates that this simulation is operating in Single Player mode, which is a Host that accepts no connections.

6.260.3.17 LatestServerTick

```
abstract Tick LatestServerTick [get]
```

latest tick on server we are aware of

6.260.3.18 LocalAddress

```
NetAddress LocalAddress [get]
```

Bound Address of the internal socket

6.260.3.19 LocalAlpha

```
float LocalAlpha [get]
```

6.260.3.20 LocalPlayer

```
abstract PlayerRef LocalPlayer [get]
```

Get LocalPlayer [PlayerRef](#)

6.260.3.21 Mode

```
SimulationModes Mode [get]
```

Gets the [SimulationModes](#) flags for The type of network peer this simulation represents.

6.260.3.22 NetConfigPointer

`NetConfig* NetConfigPointer [get]`

Current NetConfig

6.260.3.23 ObjectCount

`int ObjectCount [get]`

Returns the number of objects in the simulation.

6.260.3.24 Objects

`Dictionary<NetworkId, NetworkObjectMeta> Objects [get]`

Returns a map of all objects in the simulation.

6.260.3.25 ProjectConfig

`NetworkProjectConfig ProjectConfig [get]`

The [NetworkProjectConfig](#) file used by this [Simulation](#).

6.260.3.26 RemoteAlpha

`float RemoteAlpha [get]`

Remote Interpolation Alpha

6.260.3.27 RemoteTick

`Tick RemoteTick [get]`

Remote [Tick](#)

6.260.3.28 RemoteTickPrevious

`Tick` `RemoteTickPrevious` [get]

Remote previous `Tick`

6.260.3.29 SendDelta

`double` `SendDelta` [get]

The packet send delta time

6.260.3.30 SendRate

`int` `SendRate` [get]

The packet send rate

6.260.3.31 Stage

`SimulationStages` `Stage` [get]

Gets the current `SimulationStages` value.

6.260.3.32 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`).

6.260.3.33 TickDeltaDouble

`double` `TickDeltaDouble` [get]

The delta time of each tick as a double

6.260.3.34 TickDeltaFloat

```
float TickDeltaFloat [get]
```

The delta time of each tick as a float

6.260.3.35 TickPrevious

```
Tick TickPrevious [get]
```

The previous tick

6.260.3.36 TickRate

```
int TickRate [get]
```

The current tick rate of the simulation

6.260.3.37 TickStride

```
int TickStride [get]
```

How large the ticks the current simulation takes are

6.260.3.38 Time

```
double Time [get]
```

The current simulation time in seconds

6.260.3.39 Topology

```
Topologies Topology [get]
```

Indicates if a Server/Client or Shared Mode (relay server) topology is being used.

6.261 Simulation.AreaOfInterest Struct Reference

Area of Interest Definition

Static Public Member Functions

- static int int int z **ClampCellCoords** (int **x**, int **y**, int **z**)
Get the size of each cell in the AOI grid.
- static int **GetCellSize** ()
Convert a sphere into a set of AOI cells.
- static void **SphereToCells** (Vector3 position, float radius, HashSet< int > cells)
Convert a position into the respective cell index.
- 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 cell index into the respective cell center position.
- static Vector3 **ToCellCenter** (int index)
Get the size of the AOI grid.
- static int int int z **ToCellCoords** (int index)
Get the size of each cell in the AOI grid.
- 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 **y**

6.261.1 Detailed Description

Area of Interest Definition

6.261.2 Member Function Documentation

6.261.2.1 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.

6.261.2.2 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

6.261.2.3 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

6.261.2.4 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

6.261.2.5 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

6.261.3 Member Data Documentation

6.261.3.1 CELL_SIZE

```
int CELL_SIZE = SIZE_DEFAULT [static]
```

Size of each cell in the AOI grid.

6.261.3.2 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

6.262 SimulationBehaviour Class Reference

Base class for a [Fusion](#) aware [Behaviour](#) (derived from `UnityEngine.MonoBehaviour`). Objects derived from this object can be associated with a [NetworkRunner](#) and [Simulation](#). If a parent [NetworkObject](#) is found, this component will also be associated with that network entity.

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

6.262.1 Detailed Description

Base class for a [Fusion](#) aware [Behaviour](#) (derived from `UnityEngine.MonoBehaviour`). Objects derived from this object can be associated with a [NetworkRunner](#) and [Simulation](#). If a parent [NetworkObject](#) is found, this component will also be associated with that network entity.

6.262.2 Member Function Documentation

6.262.2.1 FixedUpdateNetwork()

```
virtual void FixedUpdateNetwork ( ) [virtual]
```

[Fusion](#) FixedUpdate timing callback.

Reimplemented in [NetworkBehaviour](#).

6.262.2.2 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](#).

6.262.3 Property Documentation

6.262.3.1 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.

6.262.3.2 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.

6.262.3.3 Object

`NetworkObject Object [get]`

The `NetworkObject` this component is associated with.

6.262.3.4 Runner

`NetworkRunner Runner [get]`

The `NetworkRunner` this component is associated with.

6.263 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

6.263.1 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)]
```

6.263.2 Property Documentation

6.263.2.1 Modes

`SimulationModes Modes [get], [set]`

Flag for which indicated peers in `SimulationModes` will execute this script.

6.263.2.2 Stages

`SimulationStages Stages [get], [set]`

Flag for which stages of the simulation loop this component will execute this script.

6.263.2.3 Topologies

`Topologies Topologies [get], [set]`

Flag for which topologies this script will execute in

6.264 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.

6.264.1 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.

6.264.2 Member Function Documentation

6.264.2.1 Dispose()

```
void Dispose ( )
```

Dispose unmanaged resources.

6.265 SimulationConfig Class Reference

Project configuration settings specific to how the [Simulation](#) class behaves.

Public Types

- enum class [DataConsistency](#)
- enum class [InputTransferModes](#)

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](#)
Features to enabled to replication such as area of interest, etc.
- [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

6.265.1 Detailed Description

Project configuration settings specific to how the [Simulation](#) class behaves.

6.265.2 Member Enumeration Documentation

6.265.2.1 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).

6.265.2.2 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.

6.265.3 Member Data Documentation

6.265.3.1 HostMigration

```
bool HostMigration
```

If, in host mode, we should allow host migration if the current host leaves.

6.265.3.2 InputDataWordCount

```
int InputDataWordCount
```

Input Data Word Count

6.265.3.3 InputTransferMode

```
InputTransferModes InputTransferMode
```

The way which input is transferred

6.265.3.4 ObjectDataConsistency

```
DataConsistency ObjectDataConsistency
```

How the server chooses to send [NetworkObject](#) updates to balance consistency and bandwidth consumption.

6.265.3.5 PlayerCount

```
int PlayerCount = 10
```

The default number of players allowed to join a game instance. Can also be changed in code when starting [Fusion](#).

6.265.3.6 ReplicationFeatures

```
NetworkProjectConfig.ReplicationFeatures ReplicationFeatures = NetworkProjectConfig.ReplicationFeatures.Schedu
```

Features to enabled to replication such as area of interest, etc.

6.265.3.7 TickRateSelection

```
TickRate.Selection TickRateSelection = TickRate.Default
```

The default tick rate to use. Can also be changed in code when starting [Fusion](#).

6.265.3.8 Topology

[Topologies](#) Topology

The topology used

6.265.4 Property Documentation

6.265.4.1 AreaOfInterestEnabled

bool AreaOfInterestEnabled [get]

Signal if AOI is enabled

6.265.4.2 InputTotalWordCount

int InputTotalWordCount [get]

6.265.4.3 SchedulingEnabled

bool SchedulingEnabled [get]

Signal if scheduling is enabled

6.265.4.4 SchedulingWithoutAOI

bool SchedulingWithoutAOI [get]

Signal if scheduling is running without AOI

6.266 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.

6.266.1 Detailed Description

[Simulation](#) Input

6.266.2 Member Function Documentation

6.266.2.1 Clear()

```
void Clear (
    int wordCount )
```

Clear a total of wordCount words from this input.

Parameters

<code>wordCount</code>	Word count to clear.
------------------------	----------------------

6.266.2.2 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.

6.266.3 Property Documentation

6.266.3.1 Data

`int* Data [get]`

Data for this input.

6.266.3.2 Header

`SimulationInputHeader* Header [get]`

Header for this input.

6.266.3.3 Player

`PlayerRef Player [get], [set]`

Player that owns this input.

6.266.3.4 Sent

`int Sent [get], [set]`

Simulation input sent count.

6.267 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.

6.267.1 Detailed Description

Buffer for SimulationInputs.

6.267.2 Constructor & Destructor Documentation

6.267.2.1 Buffer()

```
Buffer (
    NetworkProjectConfig cfg )
```

Creates a new Buffer.

Parameters

<i>cfg</i>	Network project configuration.
------------	--------------------------------

6.267.3 Member Function Documentation

6.267.3.1 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.

6.267.3.2 Clear()

```
void Clear ( )
```

Clears the buffer.

6.267.3.3 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*.

6.267.3.4 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.

6.267.3.5 Get()

```
SimulationInput Get (
    Tick tick )
```

Gets the input for *tick*.

Parameters

<i>tick</i>	Tick to get input for.
-------------	------------------------

Returns

Input for *tick*.

6.267.3.6 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*.

6.267.3.7 GetLastUsedInputHeader()

```
SimulationInputHeader GetLastUsedInputHeader ( )
```

Retrieves the last used input header data.

Returns

The last used input header data.

6.267.3.8 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.

6.267.4 Property Documentation

6.267.4.1 Count

```
int Count [get]
```

Number of inputs in the buffer.

6.267.4.2 Full

```
bool Full [get]
```

Whether the buffer is full.

6.268 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.

6.268.1 Detailed Description

[Simulation](#) Input Header

6.268.2 Member Data Documentation

6.268.2.1 [InterpAlpha](#)

`float InterpAlpha`

Interpolation alpha of the input.

6.268.2.2 [InterpFrom](#)

`Tick InterpFrom`

Interpolation from tick of the input.

6.268.2.3 InterpTo

`Tick` `InterpTo`

Interpolation to tick of the input.

6.268.2.4 SIZE

```
const int SIZE = WORD_COUNT * Allocator.REPLICATE_WORD_SIZE [static]
```

Size of the header.

6.268.2.5 Tick

`Tick` `Tick`

`Tick` of the input.

6.268.2.6 WORD_COUNT

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

Word count of the header.

6.269 SimulationMessage Struct Reference

[Simulation](#) Message

Inherits ILogDumpable.

Public Member Functions

- void ILogDumpable. **Dump** (StringBuilder builder)

Get if a flag is set on this SimulationMessage
- bool **GetFlag** (int flag)

Signal if this SimulationMessage is Targeted
- 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

6.269.1 Detailed Description

`Simulation` Message

6.269.2 Member Function Documentation

6.269.2.1 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`

6.269.2.2 CanAllocateUserPayload()

```
static bool CanAllocateUserPayload (
    int capacityInBytes ) [static]
```

Checks if a message with given size can be allocated.

6.269.2.3 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](#)

6.269.2.4 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](#)

6.269.2.5 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

6.269.2.6 IsTargeted()

```
bool IsTargeted ( )
```

Signal if this [SimulationMessage](#) is Targeted

Returns

True if this [SimulationMessage](#) is Targeted

6.269.2.7 ReadInt()

```
static int ReadInt (
    SimulationMessage * message ) [static]
```

Read a int from a [SimulationMessage](#)

Parameters

<i>message</i>	SimulationMessage to read from
----------------	--

Returns

int read

6.269.2.8 ReadNetworkedObjectRef()

```
static NetworkId ReadNetworkedObjectRef (
    SimulationMessage * message ) [static]
```

Read a [NetworkId](#) from a [SimulationMessage](#)

Parameters

<i>message</i>	SimulationMessage to read from
----------------	--

Returns

[NetworkId](#) read

6.269.2.9 ReadVector3()

```
static Vector3 ReadVector3 (
    SimulationMessage * message ) [static]
```

Read a Vector3 from a [SimulationMessage](#)

Parameters

<code>message</code>	<code>SimulationMessage</code> to read from
----------------------	---

Returns`Vector3` read**6.269.2.10 ReferenceCountAdd()**`void ReferenceCountAdd ()`Add a reference to this `SimulationMessage`**6.269.2.11 ReferenceCountSub()**`bool ReferenceCountSub ()`Subtract a reference from this `SimulationMessage`**Returns**

True if the reference count is now 0

6.269.2.12 SetDummy()`void SetDummy ()`Set this `SimulationMessage` as Dummy**6.269.2.13 SetNotTickAligned()**`void SetNotTickAligned ()`Set this `SimulationMessage` as Not Tick Aligned**6.269.2.14 SetStatic()**`void SetStatic ()`Set this `SimulationMessage` as Static**6.269.2.15 SetTarget()**`void SetTarget (`
`PlayerRef target)`Set the player target of this `SimulationMessage`

Parameters

<i>target</i>	Target Player
---------------	---------------

6.269.2.16 SetUnreliable()

```
void SetUnreliable ( )
```

Set this [SimulationMessage](#) as Unreliable

6.269.2.17 ToString() [1/2]

```
override string ToString ( )
```

[Simulation](#) Message ToString

6.269.2.18 ToString() [2/2]

```
string ToString ( 
    bool useBrackets )
```

[Simulation](#) Message ToString

6.269.2.19 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

6.269.2.20 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

6.269.2.21 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

6.269.3 Member Data Documentation

6.269.3.1 Capacity

```
int Capacity
```

Capacity in Bits of this [SimulationMessage](#)

6.269.3.2 FLAG_DUMMY

```
const int FLAG_DUMMY = 1 << 8 [static]
```

Flag for dummy messages.

6.269.3.3 FLAG_INTERNAL

```
const int FLAG_INTERNAL = 1 << 6 [static]
```

Flag for internal messages.

6.269.3.4 FLAG_NOT_TICK_ALIGNED

```
const int FLAG_NOT_TICK_ALIGNED = 1 << 7 [static]
```

Flag for messages that are not tick aligned.

6.269.3.5 FLAG_REMOTE

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

Flag for remote messages.

6.269.3.6 FLAG_STATIC

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

Flag for static messages.

6.269.3.7 FLAG_TARGET_PLAYER

```
const int FLAG_TARGET_PLAYER = 1 << 4 [static]
```

Flag for targeted messages to a player.

6.269.3.8 FLAG_TARGET_SERVER

```
const int FLAG_TARGET_SERVER = 1 << 5 [static]
```

Flag for targeted messages to the server.

6.269.3.9 FLAG_UNRELIABLE

```
const int FLAG_UNRELIABLE = 1 << 3 [static]
```

Flag for unreliable messages.

6.269.3.10 FLAG_USER_FLAGS_START

```
const int FLAG_USER_FLAGS_START = 1 << 16 [static]
```

Flag for user flags.

6.269.3.11 FLAG_USER_MESSAGE

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

Flag for user messages.

6.269.3.12 Flags

```
int Flags
```

Flags

6.269.3.13 FLAGS_RESERVED

```
const int FLAGS_RESERVED = 0xFFFF [static]
```

Flag for reserved flags.

6.269.3.14 FLAGS_RESERVED_BITS

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

Flag for reserved bits.

6.269.3.15 MAX_PAYLOAD_SIZE

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

Max user message size in bytes.

6.269.3.16 Offset

```
int Offset
```

Current offset in Bits

6.269.3.17 References

```
int References
```

Reference Count

6.269.3.18 SIZE

```
const int SIZE = 28 [static]
```

SimulationMessage size in bytes.

6.269.3.19 Source

```
PlayerRef Source
```

Source Player of this SimulationMessage

6.269.3.20 Target

```
PlayerRef Target
```

Target Player of this SimulationMessage

6.269.3.21 Tick

int **Tick**

Tick of this [SimulationMessage](#)

6.269.4 Property Documentation

6.269.4.1 IsUnreliable

bool **IsUnreliable** [get]

Signal if this [SimulationMessage](#) is Unreliable

6.270 SimulationMessagePtr Struct Reference

[Simulation](#) Message Pointer

Public Attributes

- [SimulationMessage](#) * **Message**
Pointer to the message.

6.270.1 Detailed Description

[Simulation](#) Message Pointer

6.270.2 Member Data Documentation

6.270.2.1 Message

[SimulationMessage](#)* **Message**

Pointer to the message.

6.271 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)

6.271.1 Detailed Description

Stores the runtime configuration of the simulation

6.271.2 Member Data Documentation

6.271.2.1 HostPlayer

`PlayerRef HostPlayer`

Current master client (in shared mode)

6.271.2.2 MasterClient

`PlayerRef MasterClient`

Current master client (in shared mode)

6.271.2.3 PlayerMaxCount

`int PlayerMaxCount`

Current player count

6.271.2.4 ServerMode

`SimulationModes ServerMode`

Current `Simulation` Mode

6.271.2.5 TickRate

`TickRate.Resolved TickRate`

Current tick rates and send rates for server and client

6.271.2.6 Topology

`Topologies Topology`

Current master client (in shared mode)

6.272 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 >`.

Public Member Functions

- bool `Equals (NetAddress other)`
- override bool `Equals (object obj)`
- override int `GetHashCode ()`
- override string `ToString ()`

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 Relyed connection
- bool `IsValid` [get]
Signal if this `NetAddress` is not default/empty

6.272.1 Detailed Description

Represents a Network Address, which includes a IP and Port This can contains either a IPv4 or a IPv6 address

6.272.2 Member Function Documentation

6.272.2.1 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

<code>port</code>	Port used to build the <code>NetAddress</code>
-------------------	--

Returns

New `NetAddress` reference

6.272.2.2 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

6.272.2.3 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

6.272.2.4 FromActorId()

```
static NetAddress FromActorId (
    int actorId ) [static]
```

Build a new [NetAddress](#) based on an ActorId

Parameters

<i>actor</i> ↪ <i>Id</i>	ActorId used to build the NetAddress
-----------------------------	--

Returns

Relay [NetAddress](#) that references the ActorId

ActorId must be 0 or greater

6.272.2.5 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

6.272.2.6 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

6.272.3 Property Documentation

6.272.3.1 ActorId

```
int ActorId [get]
```

Retrieves the Remote Actor ID which this [NetAddress](#) Represents

6.272.3.2 HasAddress

```
bool HasAddress [get]
```

Signal if this [NetAddress](#) has a valid IP Address

6.272.3.3 IsIPv4

```
bool IsIPv4 [get]
```

Signal if the [NetAddress](#) represents an IPv4 Address

6.272.3.4 IsIPv6

```
bool IsIPv6 [get]
```

Signal if the [NetAddress](#) represents an IPv6 Address

6.272.3.5 IsRelayAddr

```
bool IsRelayAddr [get]
```

Signal if the [NetAddress](#) is a Relayed connection

6.272.3.6 IsValid

```
bool IsValid [get]
```

Signal if this [NetAddress](#) is not default/empty

6.273 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**

6.273.1 Detailed Description

Represents a linked list of Fusion.Sockets.NetBitBuffer

6.273.2 Member Function Documentation

6.273.2.1 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
-------------	---------------------------------

6.273.2.2 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
-------------	---------------------------------

6.273.2.3 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

6.273.2.4 Remove()

```
void Remove (
    NetBitBuffer * item )
```

Remove a specific Fusion.Sockets.NetBitBuffer from the list

Parameters

<i>item</i>	NetBitBuffer to remove
-------------	------------------------

6.273.2.5 RemoveHead()

```
NetBitBuffer* RemoveHead ( )
```

Removes the first element of the list

Returns

NetBitBuffer reference

6.274 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](#)

6.274.1 Detailed Description

Accepted Command, sent by the server when a remote client connection is accepted

6.275 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

6.275.1 Detailed Description

Connect Command used to signal a remote server that a client is trying to connect to it

6.276 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

6.276.1 Detailed Description

Disconnect Command, it can be used by either side of the connection

6.277 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

6.277.1 Detailed Description

Network Command Header Describe its type and usual settings for all commands

6.277.2 Member Function Documentation

6.277.2.1 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

6.278 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

6.278.1 Detailed Description

Refuse Command, sent by the server when the connection was refused. This happens when the server has reached its max connection capacity.

6.279 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 [DefaultMtu](#)
default Maximum Transmission Unit
- 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`

6.279.1 Detailed Description

General configuration used to drive the behavior of the Socket library

6.279.2 Member Data Documentation

6.279.2.1 Address

`NetAddress Address`

Network Address used to bind the internal Socket

6.279.2.2 ConnectAttempts

`int ConnectAttempts`

Number of Connection Attempts tried by the peer before cancel the connection

6.279.2.3 ConnectInterval

`double ConnectInterval`

Interval in Seconds between attempts to connect to a remote server

6.279.2.4 ConnectionDefaultRtt

`double ConnectionDefaultRtt`

Initial RTT

6.279.2.5 ConnectionGroups

```
int ConnectionGroups
```

Number of Connection Groups supported by the local instance

6.279.2.6 ConnectionPingInterval

```
double ConnectionPingInterval
```

Interval in Seconds between ping being sent to a remote end

6.279.2.7 ConnectionSendBuffers

```
int ConnectionSendBuffers
```

Pre-allocated number of data buffers used to send data

6.279.2.8 ConnectionShutdownTime

```
double ConnectionShutdownTime
```

Timeout in Seconds to allow a disconnected Connection to be released from the Group Mapping

NetPeerGroup.UpdateShutdown

6.279.2.9 ConnectionTimeout

```
double ConnectionTimeout
```

Connection Timeout in seconds

6.279.2.10 DefaultMtu

```
int DefaultMtu
```

default Maximum Transmission Unit

6.279.2.11 MaxConnections

```
int MaxConnections
```

Max Number of Connections supported by the local instance

6.279.2.12 Notify

```
NetConfigNotify Notify
```

Package acknowledgment system configuration

6.279.2.13 OperationExpireTime

```
double OperationExpireTime
```

Max Allowed time for the Send and Receive operations, in milliseconds

6.279.2.14 PacketSize

```
int PacketSize
```

UDP Packet Size in Bytes

6.279.2.15 Simulation

```
NetConfigSimulation Simulation
```

Network simulation system configuration

6.279.2.16 SocketRecvBuffer

```
int SocketRecvBuffer
```

Size of the internal Socket receive buffer

6.279.2.17 SocketSendBuffer

int SocketSendBuffer

Size of the internal Socket send buffer

6.279.3 Property Documentation

6.279.3.1 ConnectionsPerGroup

int ConnectionsPerGroup [get]

Max number of Connection per Group based on the [ConnectionGroups](#) and [MaxConnections](#)

6.279.3.2 Defaults

[NetConfig](#) Defaults [static], [get]

Builds a [NetConfig](#) with the default values

6.279.3.3 PacketSizeInBits

int PacketSizeInBits [get]

UDP Packet Size in Bits based on [PacketSize](#)

6.280 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 Pv4AddrEqualityComparer() [get]

6.280.1 Detailed Description

Stores Addresses of a STUN Server

6.281 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
- 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 this peer, when running in Server/Host Mode
- **NetworkSceneInfo? Scene**

Scene that will be set as the starting Scene when running in Server/Host Mode
- **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.

6.281.1 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>

6.281.2 Member Function Documentation

6.281.2.1 `ToString()`

```
override string ToString ( )
```

[StartGameArgs](#) `ToString()`

6.281.3 Member Data Documentation

6.281.3.1 `Address`

[NetAddress?](#) `Address`

Peer Binding Address

Default: `NetAddress.Any(ushort)`

6.281.3.2 `AuthValues`

`AuthenticationValues` `AuthValues`

Custom Authentication Data

Default: null (default authentication values)

6.281.3.3 `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-project-config>

6.281.3.4 `ConnectionToken`

```
byte [ ] ConnectionToken
```

Connection token sent by client to server. Not used in shared mode.

Default: null (empty connection token)

6.281.3.5 CustomCallbackInterfaces

Type [] CustomCallbackInterfaces

User defined callback interfaces we will provide O(1) constant time lookup for

Default: null

6.281.3.6 CustomLobbyName

string CustomLobbyName

Session Custom Lobby to be published in

Default: null (default Lobby for each Session Type, LobbyClientServer or LobbyShared)

6.281.3.7 CustomPhotonAppSettings

FusionAppSettings CustomPhotonAppSettings

Custom Photon Application Settings

Default: null (Global PhotonAppSettings)

6.281.3.8 CustomPublicAddress

NetAddress? CustomPublicAddress

Custom Public Reflexive Address

Default: null

6.281.3.9 CustomSTUNServer

string CustomSTUNServer

Specify a Custom STUN Server used to Resolve the peer Reflexive Addresses

Default: null (no custom STUN Server)

6.281.3.10 DisableNATPunchthrough

bool DisableNATPunchthrough

Flag to disable the NAT Punchthrough implementation and connect only via Relay

Default: false

6.281.3.11 EnableClientSessionCreation

bool? EnableClientSessionCreation

Enables the Session creation when starting a Client with an specific Session Name

Default: false (clients *can not* create new Sessions)

6.281.3.12 GameMode

GameMode GameMode

[Fusion.GameMode](#) in which this peer will start

6.281.3.13 HostMigrationResume

Action<[NetworkRunner](#)> HostMigrationResume

Callback invoked when the new Host is migrating from the old Host state

Default: null

6.281.3.14 HostMigrationToken

[HostMigrationToken](#) HostMigrationToken

Host Migration Token used when restarting the [Fusion Simulation](#)

Default: null

6.281.3.15 IsOpen

bool? IsOpen

Session should be created Open or Closed to accept joins

Default: true

6.281.3.16 IsVisible

bool? IsVisible

Session should be Visible or not in the Session Lobby list

Default: true

6.281.3.17 MatchmakingMode

MatchmakingMode? MatchmakingMode

Session Join Matchmaking Mode when joining a Session. For more information, check Fusion.Photon.Realtime.[MatchmakingMode](#)

Default: MatchmakingMode.FillRoom

6.281.3.18 ObjectInitializer

[INetworkObjectInitializer](#) ObjectInitializer

See [INetworkRunnerUpdater](#)

6.281.3.19 ObjectProvider

[INetworkObjectProvider](#) ObjectProvider

Object pool to use

Default: null

6.281.3.20 OnGameStarted

Action<[NetworkRunner](#)> OnGameStarted

Callback that is invoked when the [Fusion](#) has fully started

Default: null

6.281.3.21 PlayerCount

int? PlayerCount

Number of players allowed to connect to this peer, when running in Server/Host Mode

Default: DefaultPlayers from the Global [NetworkProjectConfig](#)

6.281.3.22 Scene

[NetworkSceneInfo](#)? Scene

Scene that will be set as the starting Scene when running in Server/Host Mode

Default: null (no scene set)

6.281.3.23 SceneManager

`INetworkSceneManager` SceneManager

See [INetworkSceneManager](#).

Default: null

More about Scene Loading: <https://doc.photonengine.com/en-us/fusion/current/manual/scene-load>

6.281.3.24 SessionName

`string` SessionName

Photon Cloud Session Name used either to Create or Join a Session.

Default: null (random session matching)

6.281.3.25 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)

6.281.3.26 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)

6.281.3.27 StartGameCancellationToken

`CancellationToken` StartGameCancellationToken

Optional CancellationToken used to cancel the [NetworkRunner](#) start up process and shutdown

Defaults: null

6.281.3.28 Updater

`INetworkRunnerUpdater` Updater

See [INetworkRunnerUpdater](#)

6.281.3.29 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: false

6.281.3.30 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)

6.282 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

6.282.1 Detailed Description

Represents the result of starting the [Fusion Simulation](#)

6.282.2 Member Function Documentation

6.282.2.1 `ToString()`

```
override string ToString ( )
```

String representation of the `StartGameResult`

6.282.3 Property Documentation

6.282.3.1 `ErrorMessage`

```
string ErrorMessage [get]
```

Custom Error Message filled with data about the Shutdown. Usually used to store custom data when the `StartGame` fails.

6.282.3.2 `Ok`

```
bool Ok [get]
```

Signal if the Start was OK

6.282.3.3 `ShutdownReason`

```
ShutdownReason ShutdownReason [get]
```

Start Game Shutdown Reason

6.282.3.4 `StackTrace`

```
string StackTrace [get]
```

Optional Exception StackTrace

6.283 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.

6.283.1 Detailed Description

Represents a fusion statistics manager.

6.283.2 Property Documentation

6.283.2.1 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.

6.283.2.2 ObjectStatisticsManager

`NetworkObjectStatisticsManager ObjectStatisticsManager [get]`

Manages the network object statistics.

6.284 FusionStatisticsSnapshot Class Reference

Represents a snapshot of [Fusion](#) statistics.

Properties

- int `ForwardTicks` [get]
 Gets or sets the number of forward ticks.
- int `GeneralAllocMemoryUsedInBytes` [get]
 The number of 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 `ObjectsAllocMemoryUsedInBytes` [get]
 The number of segments allocated for objects on the simulation.
- float `OutBandwidth` [get]
 Gets or sets the outbandwidth 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.

6.284.1 Detailed Description

Represents a snapshot of [Fusion](#) statistics.

6.284.2 Property Documentation

6.284.2.1 ForwardTicks

```
int ForwardTicks [get]
```

Gets or sets the number of forward ticks.

6.284.2.2 GeneralAllocMemoryUsedInBytes

```
int GeneralAllocMemoryUsedInBytes [get]
```

The number of segments allocated for general purposes on the simulation.

6.284.2.3 InBandwidth

```
float InBandwidth [get]
```

Gets or sets the in-bandwidth value.

6.284.2.4 InObjectUpdates

```
int InObjectUpdates [get]
```

The number of received objects updates per packet.

6.284.2.5 InPackets

```
int InPackets [get]
```

Gets or sets the number of incoming packets.

6.284.2.6 InputInBandwidth

```
float InputInBandwidth [get]
```

Gets the input in bandwidth.

6.284.2.7 InputOutBandwidth

```
float InputOutBandwidth [get]
```

Gets the input out bandwidth.

6.284.2.8 InputReceiveDelta

```
float InputReceiveDelta [get]
```

Gets the Input Receive Delta.

6.284.2.9 InterpolationOffset

```
float InterpolationOffset [get]
```

Gets the Interpolation Offset.

6.284.2.10 InterpolationSpeed

```
float InterpolationSpeed [get]
```

Gets the Interpolation Speed.

6.284.2.11 ObjectsAllocMemoryUsedInBytes

```
int ObjectsAllocMemoryUsedInBytes [get]
```

The number of segments allocated for objects on the simulation.

6.284.2.12 OutBandwidth

```
float OutBandwidth [get]
```

Gets or sets the outbandwith property.

6.284.2.13 OutObjectUpdates

```
int OutObjectUpdates [get]
```

The number of sent objects updates per packet.

6.284.2.14 OutPackets

```
int OutPackets [get]
```

Gets or sets the number of outgoing packets.

6.284.2.15 Resimulations

```
int Resimulations [get]
```

Gets or sets the number of resimulations.

6.284.2.16 RoundTripTime

```
float RoundTripTime [get]
```

Gets or sets the round trip time (RTT) in seconds.

6.284.2.17 SimulationSpeed

```
float SimulationSpeed [get]
```

Gets the [Simulation Speed](#).

6.284.2.18 SimulationTimeOffset

```
float SimulationTimeOffset [get]
```

Gets the [Simulation Time Offset](#).

6.284.2.19 StateReceiveDelta

```
float StateReceiveDelta [get]
```

Gets the State Receive Delta.

6.284.2.20 TimeResets

```
int TimeResets [get]
```

Gets the Time Resets count.

6.285 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 \(NetworkId id, out NetworkObjectStatisticsSnapshot objectStatistics\)](#)
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.

6.285.1 Detailed Description

Manages network object statistics for monitored network objects.

6.285.2 Member Function Documentation

6.285.2.1 ClearMonitoredNetworkObjects()

```
void ClearMonitoredNetworkObjects ( )
```

Clears the list of monitored network objects.

6.285.2.2 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.

6.285.2.3 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.

6.286 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 outbandwidth property.
- int **OutPackets** [get]
Gets or sets the number of outgoing packets.

6.286.1 Detailed Description

Represents a snapshot of network object statistics.

6.286.2 Property Documentation

6.286.2.1 InBandwidth

```
float InBandwidth [get]
```

Gets or sets the in-bandwidth value.

6.286.2.2 InPackets

```
int InPackets [get]
```

Gets or sets the number of incoming packets.

6.286.2.3 OutBandwidth

```
float OutBandwidth [get]
```

Gets or sets the outbandwith property.

6.286.2.4 OutPackets

```
int OutPackets [get]
```

Gets or sets the number of outgoing packets.

6.287 Tick Struct Reference

A tick is a 32-bit integer that represents a frame number.

Inherits IComparable< 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.

6.287.1 Detailed Description

A tick is a 32-bit integer that represents a frame number.

6.287.2 Member Function Documentation

6.287.2.1 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.

6.287.2.2 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.

6.287.2.3 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.

6.287.2.4 GetHashCode()

```
override int GetHashCode ( )
```

Serves as the default hash function.

Returns

A hash code for the current [Tick](#).

6.287.2.5 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.

6.287.2.6 operator bool()

```
static implicit operator bool (   
    Tick value ) [static]
```

Converts a [Tick](#) to a boolean.

Parameters

<code>value</code>	The Tick to convert.
--------------------	--------------------------------------

Returns

true if the [Tick](#)'s raw value is greater than 0; otherwise, false.

6.287.2.7 operator int()

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

Converts a [Tick](#) to an integer.

Parameters

<code>value</code>	The Tick to convert.
--------------------	--------------------------------------

Returns

An integer that represents the raw value of the specified [Tick](#).

6.287.2.8 operator Tick()

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

Converts an integer to a [Tick](#).

Parameters

<code>value</code>	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.

6.287.2.9 operator"!=()

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

Determines whether the first specified [Tick](#) is not equal to the second specified [Tick](#).

6.287.2.10 operator<()

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

Determines whether the first specified [Tick](#) is less than the second specified [Tick](#).

6.287.2.11 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](#).

6.287.2.12 operator==()

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

Determines whether the first specified [Tick](#) is equal to the second specified [Tick](#).

6.287.2.13 operator>()

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

Determines whether the first specified [Tick](#) is greater than the second specified [Tick](#).

6.287.2.14 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](#).

6.287.2.15 ToString()

```
override string ToString ( )
```

String representation of the [Tick](#).

6.287.3 Member Data Documentation

6.287.3.1 ALIGNMENT

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

The alignment of the [Tick](#) structure in bytes.

6.287.3.2 Raw

```
int Raw
```

The raw value of the [Tick](#). This represents a frame number.

6.287.3.3 SIZE

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

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

6.288 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.

6.288.1 Detailed Description

Provides a mechanism for comparing two [Tick](#) objects for equality.

6.288.2 Member Function Documentation

6.288.2.1 Equals()

```
bool Equals (
    Tick x,
    Tick y )
```

Determines whether the specified [Tick](#) objects are equal.

Parameters

x	The first Tick object to compare.
y	The second Tick object to compare.

Returns

true if the specified [Tick](#) objects are equal; otherwise, false.

6.288.2.2 GetHashCode()

```
int GetHashCode (
    Tick obj )
```

Serves as a hash function for a [Tick](#) object.

Parameters

obj	The Tick object for which to get a hash code.
-----	---

Returns

A hash code for the specified [Tick](#) object.

6.289 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.

6.289.1 Detailed Description

Provides a mechanism for comparing two [Tick](#) objects.

6.289.2 Member Function Documentation

6.289.2.1 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

x	The first Tick object to compare.
y	The second Tick object to compare.

Returns

A signed integer that indicates the relative values of x and y.

6.290 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](#).

6.290.1 Detailed Description

A tick accumulator.

6.290.2 Member Function Documentation

6.290.2.1 AddTicks()

```
void AddTicks (
    int ticks )
```

Adds a specified number of ticks to the [TickAccumulator](#).

Parameters

<i>ticks</i>	The number of ticks to add.
--------------	-----------------------------

6.290.2.2 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.

6.290.2.3 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.

6.290.2.4 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.

6.290.2.5 Start()

```
void Start ( )
```

Starts the [TickAccumulator](#), allowing it to accumulate ticks.

6.290.2.6 StartNew()

```
static TickAccumulator StartNew ( ) [static]
```

Starts a new [TickAccumulator](#).

Returns

A new [TickAccumulator](#).

6.290.2.7 Stop()

```
void Stop ( )
```

Stops the [TickAccumulator](#) from accumulating ticks.

6.290.3 Property Documentation

6.290.3.1 Pending

```
int Pending [get]
```

Gets the number of pending ticks in the [TickAccumulator](#).

6.290.3.2 Remainder

```
double Remainder [get]
```

Gets the remaining time in the [TickAccumulator](#).

6.290.3.3 Running

```
bool Running [get]
```

Gets a value indicating whether the [TickAccumulator](#) is running.

6.290.3.4 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.
----------------------------------	---

6.291 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.

6.291.1 Detailed Description

A tick rate is a collection of tick rates.

6.291.2 Member Enumeration Documentation

6.291.2.1 ValidateResult

```
enum ValidateResult [strong]
```

Enumerates the possible results of validating a tick rate selection.

Enumerator

Ok	The tick rate selection is valid.
Error	An error occurred during validation.
NotFound	The tick rate selection was not found.
InvalidTickRate	The tick rate selection is invalid.
ServerIndexOutOfRange	The server index in the tick rate selection is out of range.
ClientSendIndexOutOfRange	The client send index in the tick rate selection is out of range.
ServerSendIndexOutOfRange	The server send index in the tick rate selection is out of range.
ServerSendRateLargerThanTickRate	The server send rate in the tick rate selection is larger than the tick rate.

6.291.3 Member Function Documentation

6.291.3.1 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](#).

6.291.3.2 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.
----------------------------------	---------------------------------------

6.291.3.3 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.

6.291.3.4 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.
------------------------------------	--

6.291.3.5 Init()

```
static void Init ( ) [static]
```

Initializes the [TickRate](#) class by setting up the valid tick rates and their lookup dictionary.

6.291.3.6 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.

6.291.3.7 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.

6.291.3.8 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.

6.291.3.9 ToArray()

```
int [ ] ToArray ( )
```

Converts the tick rates to an array.

Returns

An array containing the tick rates.

6.291.3.10 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.

6.291.3.11 ValidateSelection()

```
ValidateResult ValidateSelection (  
    Selection selected )
```

Validates the tick rates in the specified [Selection](#).

Parameters

<code>selected</code>	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.

6.291.4 Property Documentation

6.291.4.1 Available

```
IReadOnlyList<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.

6.291.4.2 Client

```
int Client [get]
```

Gets the tick rate for the client.

6.291.4.3 Count

```
int Count [get]
```

Gets the count of tick rates in the [TickRate](#).

6.291.4.4 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.

6.292 TickRate.Resolved Struct Reference

Represents a resolved tick rate.

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.

6.292.1 Detailed Description

Represents a resolved tick rate.

The tick rate is resolved by the client and server tick rates.

6.292.2 Member Data Documentation

6.292.2.1 Client

```
int Client
```

The tick rate for the client.

6.292.2.2 ClientSend

```
int ClientSend
```

The send tick rate for the client.

6.292.2.3 Server

```
int Server
```

The tick rate for the server.

6.292.2.4 ServerSend

```
int ServerSend
```

The send tick rate for the server.

6.292.2.5 SIZE

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

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

6.292.2.6 WORDS

```
const int WORDS = SIZE / Allocator.REPLICATE_WORD_SIZE [static]
```

The size of the [Resolved](#) structure in words.

6.292.3 Property Documentation

6.292.3.1 ClientSendDelta

```
double ClientSendDelta [get]
```

Gets the delta time for the client send rate.

6.292.3.2 ClientTickDelta

```
double ClientTickDelta [get]
```

Gets the delta time for the client tick rate.

6.292.3.3 ClientTickStride

```
int ClientTickStride [get]
```

Gets the stride of the client tick rate. This is always 1.

6.292.3.4 ServerSendDelta

```
double ServerSendDelta [get]
```

Gets the delta time for the server send rate.

6.292.3.5 ServerTickDelta

```
double ServerTickDelta [get]
```

Gets the delta time for the server tick rate.

6.292.3.6 ServerTickStride

```
int ServerTickStride [get]
```

Gets the stride of the server tick rate relative to the client tick rate.

6.293 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.

6.293.1 Detailed Description

Represents a selection of tick rates for client and server.

6.293.2 Member Data Documentation

6.293.2.1 Client

```
int Client
```

The tick rate for the client.

6.293.2.2 ClientSendIndex

```
int ClientSendIndex
```

The index of the client's send tick rate in the available tick rates.

6.293.2.3 ServerIndex

```
int ServerIndex
```

The index of the server's tick rate in the available tick rates.

6.293.2.4 ServerSendIndex

```
int ServerSendIndex
```

The index of the server's send tick rate in the available tick rates.

6.294 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`.

6.294.1 Detailed Description

A timer that is based on ticks instead of seconds.

6.294.2 Member Function Documentation

6.294.2.1 CreateFromSeconds()

```
static TickTimer CreateFromSeconds (
    NetworkRunner runner,
    float delayInSeconds ) [static]
```

Creates a `TickTimer` from a specified delay in seconds.

Parameters

<code>runner</code>	The <code>NetworkRunner</code> associated with the <code>TickTimer</code> .
<code>delayInSeconds</code>	The delay in seconds to set the <code>TickTimer</code> 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`.

6.294.2.2 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](#).

6.294.2.3 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.

6.294.2.4 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.

6.294.2.5 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.

6.294.2.6 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.

6.294.2.7 ToString()

```
override string ToString ( )
```

Returns a string that represents the current [TickTimer](#).

Returns

A string that represents the current [TickTimer](#).

6.294.3 Property Documentation

6.294.3.1 IsRunning

```
bool IsRunning [get]
```

Gets a value indicating whether the [TickTimer](#) is running.

true if the [TickTimer](#) is running; otherwise, false.

6.294.3.2 None

```
TickTimer None [static], [get]
```

Gets a [TickTimer](#) that is not running.

6.294.3.3 TargetTick

```
int? TargetTick [get]
```

Gets the target tick of the [TickTimer](#).

The target tick if the [TickTimer](#) is running; otherwise, null.

6.295 TimeSyncConfiguration Class Reference

Time Synchronization Configuration

Public Attributes

- double [MaxLateInputs](#) = 5.0
- double [MaxLateSnapshots](#) = 5.0
- int [RedundantInputs](#) = 2
- int [RedundantSnapshots](#) = 2
- double [SampleWindowSeconds](#) = 1.0

6.295.1 Detailed Description

Time Synchronization Configuration

6.295.2 Member Data Documentation

6.295.2.1 MaxLateInputs

```
double MaxLateInputs = 5.0
```

The maximum percentage of inputs that should ever arrive late because of jitter.

Decreasing this increases the client's simulation offset.

6.295.2.2 MaxLateSnapshots

```
double MaxLateSnapshots = 5.0
```

The maximum percentage of snapshots that should ever arrive late because of jitter.

Decreasing this increases the client's interpolation delay.

6.295.2.3 RedundantInputs

```
int RedundantInputs = 2
```

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.

6.295.2.4 RedundantSnapshots

```
int RedundantSnapshots = 2
```

The number of consecutive snapshots a client should be able to miss without disrupting their interpolation.

Increasing this increases the client's interpolation delay.

6.295.2.5 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.

6.296 UnitAttribute Class Reference

Unit Attribute class. Used to mark a field with the respective [Units](#)

Inherits DecoratingPropertyAttribute.

Public Member Functions

- **UnitAttribute** ([Units](#) units)

Properties

- [Units](#) **Unit** [get]

6.296.1 Detailed Description

Unit Attribute class. Used to mark a field with the respective [Units](#)

6.297 UnityContextMenuItemAttribute Class Reference

Unity ContextMenuItemAttribute

Inherits Attribute.

Public Member Functions

- [UnityContextMenuItemAttribute](#) (string function, string name)
ContextMenuItemAttribute Constructor

Properties

- int [order](#) [get, set]
ContextMenuItemAttribute Order

6.297.1 Detailed Description

Unity ContextMenuItemAttribute

6.297.2 Constructor & Destructor Documentation

6.297.2.1 UnityContextMenuItemAttribute()

```
UnityContextMenuItemAttribute (
    string function,
    string name )
```

ContextMenuItemAttribute Constructor

6.297.3 Property Documentation

6.297.3.1 order

```
int order [get], [set]
```

ContextMenuItemAttribute Order

6.298 UnityDelayedAttribute Class Reference

Unity DelayedAttribute

Inherits Attribute.

Properties

- int **order** [get, set]
DelayedAttribute Order

6.298.1 Detailed Description

Unity DelayedAttribute

6.298.2 Property Documentation

6.298.2.1 order

```
int order [get], [set]
```

DelayedAttribute Order

6.299 UnityFormerlySerializedAsAttribute Class Reference

Unity FormerlySerializedAsAttribute

Inherits Attribute.

Public Member Functions

- [UnityFormerlySerializedAsAttribute](#) (string oldName)
FormerlySerializedAsAttribute Constructor

6.299.1 Detailed Description

Unity FormerlySerializedAsAttribute

6.299.2 Constructor & Destructor Documentation

6.299.2.1 UnityFormerlySerializedAsAttribute()

```
UnityFormerlySerializedAsAttribute (
    string oldName )
```

FormerlySerializedAsAttribute Constructor

6.300 UnityHeaderAttribute Class Reference

Unity HeaderAttribute

Inherits Attribute.

Public Member Functions

- [UnityHeaderAttribute](#) (string header)
HeaderAttribute Constructor

Properties

- int [order](#) [get, set]
HeaderAttribute Order

6.300.1 Detailed Description

Unity HeaderAttribute

6.300.2 Constructor & Destructor Documentation

6.300.2.1 UnityHeaderAttribute()

```
UnityHeaderAttribute (
    string header )
```

HeaderAttribute Constructor

6.300.3 Property Documentation

6.300.3.1 order

```
int order [get], [set]
```

HeaderAttribute Order

6.301 UnityMinAttribute Class Reference

Unity MinAttribute

Inherits Attribute.

Public Member Functions

- [UnityMinAttribute \(float min\)](#)

MinAttribute Constructor

Properties

- int [order \[get, set\]](#)

MinAttribute Order

6.301.1 Detailed Description

Unity MinAttribute

6.301.2 Constructor & Destructor Documentation

6.301.2.1 UnityMinAttribute()

```
UnityMinAttribute ( float min )
```

MinAttribute Constructor

6.301.3 Property Documentation

6.301.3.1 order

```
int order [get], [set]
```

MinAttribute Order

6.302 UnityMultilineAttribute Class Reference

Unity MultilineAttribute

Inherits Attribute.

Properties

- int **order** [get, set]
MultilineAttribute Order

6.302.1 Detailed Description

Unity MultilineAttribute

6.302.2 Property Documentation

6.302.2.1 order

```
int order [get], [set]
```

MultilineAttribute Order

6.303 UnityNonReorderableAttribute Class Reference

Unity NonReorderableAttribute

Inherits Attribute.

Properties

- int `order` [get, set]
NonReorderableAttribute Order

6.303.1 Detailed Description

Unity NonReorderableAttribute

6.303.2 Property Documentation

6.303.2.1 `order`

`int order [get], [set]`

NonReorderableAttribute Order

6.304 UnityNonSerializedAttribute Class Reference

Unity NonSerializedAttribute

Inherits Attribute.

6.304.1 Detailed Description

Unity NonSerializedAttribute

6.305 UnityRangeAttribute Class Reference

Unity RangeAttribute

Inherits Attribute.

Public Member Functions

- [UnityRangeAttribute](#) (float min, float max)
RangeAttribute Constructor

Properties

- int [order](#) [get, set]
RangeAttribute Order

6.305.1 Detailed Description

Unity RangeAttribute

6.305.2 Constructor & Destructor Documentation

6.305.2.1 UnityRangeAttribute()

```
UnityRangeAttribute (
    float min,
    float max )
```

RangeAttribute Constructor

6.305.3 Property Documentation

6.305.3.1 order

```
int order [get], [set]
```

RangeAttribute Order

6.306 UnitySerializeField Class Reference

Unity SerializeField

Inherits Attribute.

6.306.1 Detailed Description

Unity SerializeField

6.307 UnitySerializeReference Class Reference

Unity SerializeReference

Inherits Attribute.

6.307.1 Detailed Description

Unity SerializeReference

6.308 UnitySpaceAttribute Class Reference

Unity SpaceAttribute

Inherits Attribute.

Public Member Functions

- [UnitySpaceAttribute](#) (float space)
SpaceAttribute Constructor

Properties

- int [order](#) [get, set]
SpaceAttribute Order

6.308.1 Detailed Description

Unity SpaceAttribute

6.308.2 Constructor & Destructor Documentation

6.308.2.1 UnitySpaceAttribute()

```
UnitySpaceAttribute (
    float space )
```

SpaceAttribute Constructor

6.308.3 Property Documentation

6.308.3.1 order

int order [get], [set]

SpaceAttribute Order

6.309 UnityTooltipAttribute Class Reference

Unity TooltipAttribute

Inherits Attribute.

Public Member Functions

- [UnityTooltipAttribute](#) (string tooltip)
TooltipAttribute Constructor

Properties

- int [order](#) [get, set]
TooltipAttribute Order

6.309.1 Detailed Description

Unity TooltipAttribute

6.309.2 Constructor & Destructor Documentation

6.309.2.1 UnityTooltipAttribute()

```
UnityTooltipAttribute (
    string tooltip )
```

TooltipAttribute Constructor

6.309.3 Property Documentation

6.309.3.1 order

`int order [get], [set]`

TooltipAttribute Order

6.310 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.

6.310.1 Detailed Description

Represents a compressed Vector2 value for network transmission.

6.310.2 Member Function Documentation

6.310.2.1 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.

6.310.2.2 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.

6.310.2.3 GetHashCode()

```
override int GetHashCode ( )
```

Returns the hash code for the current [Vector2Compressed](#) instance.

Returns

A hash code for the current [Vector2Compressed](#) instance.

6.310.2.4 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.

6.310.2.5 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](#).

6.310.2.6 operator"!=()

```
static bool operator!= (
    Vector2Compressed left,
    Vector2Compressed right ) [static]
```

Inequality operator for [Vector2Compressed](#) struct.

Parameters

<i>left</i>	First <code>Vector2Compressed</code> instance.
<i>right</i>	Second <code>Vector2Compressed</code> instance.

Returns

True if the value of the first `Vector2Compressed` instance is not equal to the value of the second `Vector2Compressed` instance, otherwise false.

6.310.2.7 operator==()

```
static bool operator== (
    Vector2Compressed left,
    Vector2Compressed right ) [static]
```

Equality operator for `Vector2Compressed` struct.

Parameters

<i>left</i>	First <code>Vector2Compressed</code> instance.
<i>right</i>	Second <code>Vector2Compressed</code> instance.

Returns

True if the value of the first `Vector2Compressed` instance is equal to the value of the second `Vector2Compressed` instance, otherwise false.

6.310.3 Member Data Documentation**6.310.3.1 xEncoded**

```
int xEncoded
```

Encoded value of the x component.

6.310.3.2 yEncoded

```
int yEncoded
```

Encoded value of the y component.

6.310.4 Property Documentation

6.310.4.1 X

```
float X [get], [set]
```

Gets or sets the x component.

6.310.4.2 Y

```
float Y [get], [set]
```

Gets or sets the y component.

6.311 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.

6.311.1 Detailed Description

Represents a compressed Vector3 value for network transmission.

6.311.2 Member Function Documentation

6.311.2.1 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

<code>obj</code>	The object to compare with the current <code>Vector3Compressed</code> instance.
------------------	---

Returns

True if the provided object is a `Vector3Compressed` instance and it's equal to the current `Vector3Compressed` instance, otherwise false.

6.311.2.2 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.

6.311.2.3 GetHashCode()

```
override int GetHashCode ( )
```

Returns the hash code for the current [Vector3Compressed](#) instance.

Returns

A hash code for the current [Vector3Compressed](#) instance.

6.311.2.4 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.

6.311.2.5 operator Vector3()

```
static implicit operator Vector3 (
    Vector3Compressed q ) [static]
```

Implicit conversion from [Vector3Compressed](#) to [Vector3](#).

Parameters

<code>q</code>	The Vector3Compressed instance to convert.
----------------	--

Returns

The decompressed [Vector3](#) value of the [Vector3Compressed](#) instance.

6.311.2.6 operator Vector3Compressed() [1/2]

```
static implicit operator Vector3Compressed (
    Vector2 v ) [static]
```

Implicit conversion from [Vector2](#) to [Vector3Compressed](#).

Parameters

<code>v</code>	The Vector2 value to convert.
----------------	---

Returns

A new [Vector3Compressed](#) instance with the compressed value of the [Vector2](#).

6.311.2.7 operator Vector3Compressed() [2/2]

```
static implicit operator Vector3Compressed (
    Vector3 v ) [static]
```

Implicit conversion from [Vector3](#) to [Vector3Compressed](#).

Parameters

<code>v</code>	The Vector3 value to convert.
----------------	---

Returns

A new [Vector3Compressed](#) instance with the compressed value of the [Vector3](#).

6.311.2.8 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.

6.311.2.9 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.

6.311.3 Member Data Documentation

6.311.3.1 xEncoded

```
int xEncoded
```

Encoded value of the x component.

6.311.3.2 yEncoded

```
int yEncoded
```

Encoded value of the y component.

6.311.3.3 zEncoded

```
int zEncoded
```

Encoded value of the z component.

6.311.4 Property Documentation

6.311.4.1 X

```
float X [get], [set]
```

Gets or sets the x component.

6.311.4.2 Y

```
float Y [get], [set]
```

Gets or sets the y component.

6.311.4.3 Z

```
float Z [get], [set]
```

Gets or sets the z component.

6.312 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.

6.312.1 Detailed Description

Represents a compressed Vector4 value for network transmission.

6.312.2 Member Function Documentation

6.312.2.1 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.

6.312.2.2 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.

6.312.2.3 GetHashCode()

```
override int GetHashCode ( )
```

Returns the hash code for the current [Vector4Compressed](#) instance.

Returns

A hash code for the current [Vector4Compressed](#) instance.

6.312.2.4 operator Vector4()

```
static implicit operator Vector4 (
    Vector4Compressed q ) [static]
```

Implicit conversion from `Vector4Compressed` to `Vector4`.

Parameters

<code>q</code>	The <code>Vector4Compressed</code> instance to convert.
----------------	---

Returns

The decompressed `Vector4` value of the `Vector4Compressed` instance.

6.312.2.5 operator Vector4Compressed()

```
static implicit operator Vector4Compressed (
    Vector4 v ) [static]
```

Implicit conversion from `Vector4` to `Vector4Compressed`.

Parameters

<code>v</code>	The <code>Vector4</code> value to convert.
----------------	--

Returns

A new `Vector4Compressed` instance with the compressed value of the `Vector4`.

6.312.2.6 operator"!=()

```
static bool operator!= (
    Vector4Compressed left,
    Vector4Compressed right ) [static]
```

Inequality operator for `Vector4Compressed` struct.

Parameters

<code>left</code>	First <code>Vector4Compressed</code> instance.
<code>right</code>	Second <code>Vector4Compressed</code> instance.

Returns

True if the value of the first `Vector4Compressed` instance is not equal to the value of the second `Vector4Compressed` instance, otherwise false.

6.312.2.7 operator==()

```
static bool operator== (
    Vector4Compressed left,
    Vector4Compressed right ) [static]
```

Equality operator for `Vector4Compressed` struct.

Parameters

<i>left</i>	First <code>Vector4Compressed</code> instance.
<i>right</i>	Second <code>Vector4Compressed</code> instance.

Returns

True if the value of the first `Vector4Compressed` instance is equal to the value of the second `Vector4Compressed` instance, otherwise false.

6.312.3 Member Data Documentation

6.312.3.1 wEncoded

```
int wEncoded
```

Encoded value of the w component.

6.312.3.2 xEncoded

```
int xEncoded
```

Encoded value of the x component.

6.312.3.3 yEncoded

```
int yEncoded
```

Encoded value of the y component.

6.312.3.4 zEncoded

```
int zEncoded
```

Encoded value of the z component.

6.312.4 Property Documentation

6.312.4.1 W

```
float W [get], [set]
```

Gets or sets the w component.

6.312.4.2 X

```
float X [get], [set]
```

Gets or sets the x component.

6.312.4.3 Y

```
float Y [get], [set]
```

Gets or sets the y component.

6.312.4.4 Z

```
float Z [get], [set]
```

Gets or sets the z component.

6.313 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 [DolfAttributeBase](#).

Public Member Functions

- **WarnIfAttribute** (string propertyPath, bool compareToValue, string message, [CompareOperator](#) compare=[CompareOperator.Equal](#))
- **WarnIfAttribute** (string propertyPath, double compareToValue, string message, [CompareOperator](#) compare=[CompareOperator.Equal](#))
- **WarnIfAttribute** (string propertyPath, long compareToValue, string message, [CompareOperator](#) compare=[CompareOperator.Equal](#))
- **WarnIfAttribute** (string propertyPath, string message)

Public Attributes

- bool **AsBox**
- string **Message**

The default warning text, when a warning is shown.

Additional Inherited Members

6.313.1 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).

Value of condition method is converted to a long. Null = 0, False = 0, True = 1, Unity Object = InstanceId

6.313.2 Member Data Documentation

6.313.2.1 Message

string **Message**

The default warning text, when a warning is shown.

6.314 WeaverGeneratedAttribute Class Reference

Weaver Generated Attribute

Inherits Attribute.

6.314.1 Detailed Description

Weaver Generated Attribute

Index

_128, [61](#)
 Data, [61](#)
 SIZE, [61](#)
_16, [62](#)
 Data, [62](#)
 SIZE, [62](#)
_2, [63](#)
 Data, [63](#)
 SIZE, [63](#)
_256, [63](#)
 Data, [64](#)
 SIZE, [64](#)
_32, [64](#)
 Data, [65](#)
 SIZE, [65](#)
_4, [65](#)
 Data, [66](#)
 SIZE, [66](#)
_512, [66](#)
 Data, [67](#)
 SIZE, [67](#)
_64, [67](#)
 Data, [67](#)
 SIZE, [68](#)
_8, [68](#)
 Data, [68](#)
 SIZE, [68](#)
_debruijnTable
 DynamicHeap, [99](#)
_reserved
 NetworkObjectHeader, [435](#)
_value0
 NetworkObjectTypeld, [456](#)
_value1
 NetworkObjectTypeld, [456](#)

AABB, [214](#)
 AABB, [214, 215](#)
 Center, [215](#)
 Extents, [215](#)
 Max, [215](#)
 Min, [215](#)
Accept
 NetworkRunnerCallbackArgs.ConnectRequest, [571](#)
ACCURACY
 ReadWriteUtils, [669](#)
Acquire
 INetworkAssetSource< T >, [171](#)
AcquirePrefabInstance

INetworkObjectProvider, [175](#)
ActivePlayers
 NetworkRunner, [562](#)
 Simulation, [727](#)
ActorId
 NetAddress, [770](#)
Add
 INetworkDictionary, [173](#)
 INetworkLinkedList, [174](#)
 NetworkDictionary< K, V >, [360](#)
 NetworkLinkedList< T >, [391](#)
 SerializableDictionary< TKey, TValue >, [709](#)
 SimulationInput.Buffer, [749](#)
AddBehaviour< T >
 Behaviour, [89](#)
AddCallbacks
 NetworkRunner, [519](#)
AddFirst
 NetBitBufferList, [772](#)
AddGlobal
 NetworkRunner, [519](#)
AddInstance
 NetworkPrefabTable, [482](#)
AdditionalJitter
 NetworkSimulationConfiguration, [596](#)
AdditionalLoss
 NetworkSimulationConfiguration, [597](#)
AddLast
 NetBitBufferList, [772](#)
AddMode
 NetworkRunnerUpdaterDefaultInvokeSettings, [577](#)
AddOnCompleted
 NetworkSceneAsyncOp, [579](#)
AddPlayerAreaOfInterest
 NetworkRunner, [519](#)
Address
 NetConfig, [778](#)
 Ptr, [655](#)
 StartGameArgs, [784](#)
AddSceneRef
 NetworkSceneInfo, [586](#)
AddSource
 NetworkPrefabTable, [482](#)
AddTicks
 TickAccumulator, [807](#)
AddTime
 TickAccumulator, [807](#)
After
 Fusion, [53](#)

AfterAllTicks
 IAfterAllTicks, 153

AfterClientPredictionReset
 IAfterClientPredictionReset, 154

AfterHostMigration
 IAfterHostMigration, 154

AfterRender
 IAfterRender, 155

AfterSimulation
 Simulation, 720

AfterSpawned
 IAfterSpawned, 155

AfterTick
 IAfterTick, 156

AfterUpdate
 IAfterUpdate, 156

Simulation, 720

ALIGNMENT
 NetworkId, 380
 NetworkObjectGuid, 429
 NetworkObjectNestingKey, 443
 NetworkObjectTypeid, 456
 NetworkPrefabId, 468
 NetworkPrefabRef, 478
 Tick, 804

ALL
 AuthorityMasks, 87

All
 Fusion, 49

Allocate
 DynamicHeapInstance, 100
 SimulationMessage, 756

AllocateArray< T >
 DynamicHeapInstance, 100

AllocateArrayPointers< T >
 DynamicHeapInstance, 101

AllocateTracked< T >
 DynamicHeapInstance, 101

AllocateTrackedArray< T >
 DynamicHeapInstance, 102

AllocateTrackedArrayPointers< T >
 DynamicHeapInstance, 102

Allocator, 69
 BUCKET_COUNT, 70
 BUCKET_INVALID, 70
 HEAP_ALIGNMENT, 70
 REPLICATE_WORD_ALIGN, 70
 REPLICATE_WORD_SHIFT, 71
 REPLICATE_WORD_SIZE, 71

Allocator.Config, 71
 BlockByteSize, 75
 BlockCount, 74
 BlockShift, 74
 BlockWordCount, 75
 Config, 72
 DEFAULT_BLOCK_COUNT, 74
 DEFAULT_BLOCK_SHIFT, 74
 Equals, 72, 73

GetHashCode, 73
 GlobalsSize, 74
 HeapSizeAllocated, 75
 HeapSizeUsable, 75
 SIZE, 74
 ToString, 73

AllowStateAuthorityOverride
 Fusion, 43

Alpha
 NetworkBehaviourBufferInterpolator, 317
 Query, 231
 QueryParams, 233
 TickAccumulator, 808

AlreadyRunning
 Fusion, 51

Angle, 75
 Clamp, 77
 Equals, 77, 78
 GetHashCode, 78
 Lerp, 78
 Max, 78
 Min, 79
 NetworkBehaviourBufferInterpolator, 309
 operator Angle, 79, 80
 operator double, 80
 operator float, 80
 operator!=, 81
 operator<, 82
 operator<=, 82
 operator>, 83
 operator>=, 84
 operator+, 81
 operator-, 81
 operator==, 83
 SIZE, 84
 ToString, 84

Animator
 NetworkMecanimAnimator, 407

Any
 NetAddress, 768

AnyIPv6
 NetAddress, 768

ApplyTiming
 NetworkMecanimAnimator, 407

AreaOfInterest
 Fusion, 43

AreaOfInterestEnabled
 SimulationConfig, 745

AreaOfInterestOverride
 NetworkTRSPData, 637

AsCustom
 NetworkObjectTypeid, 456

AsIndex
 NetworkPrefabId, 469
 PlayerRef, 647
 SceneRef, 706

AsInternalStructId
 NetworkObjectTypeid, 457

AsPathHash
SceneRef, 706

AspectRatio
NormalizedRectAttribute, 640

AsPrefabId
NetworkObjectTypeId, 457

AsSceneObjectId
NetworkObjectTypeId, 457

AssembliesToWeave
NetworkProjectConfig, 494

AssetGuid
INetworkPrefabSource, 177

AssetObject, 84

Assign
NetworkString< TSize >, 606

AssignInputAuthority
NetworkObject, 410

Attach
NetworkRunner, 519, 520

AuthenticationTicketExpired
Fusion, 51

AuthenticationValues
NetworkRunner, 562

AuthorityMasks, 87
ALL, 87
INPUT, 87
NONE, 87
PROXY, 88
STATE, 88

AuthValues
StartGameArgs, 784

AutoHostOrClient
Fusion, 42

AutoUpdateAreaOfInterestOverride
NetworkTransform, 633

Available
TickRate, 816

Awaiter
NetworkSceneAsyncOp.Awaiter, 583
NetworkSpawnOp.Awaiter, 601

Awake
NetworkObject, 410

Before
Fusion, 53

BeforeAllTicks
IBeforeAllTicks, 158

BeforeClientPredictionReset
IBeforeClientPredictionReset, 159

BeforeCopyPreviousState
IBeforeCopyPreviousState, 160

BeforeFirstTick
Simulation, 720

BeforeHitboxRegistration
IBeforeHitboxRegistration, 160

BeforeSimulation
IBeforeSimulation, 161
Simulation, 720

BeforeTick

IBeforeTick, 162

BeforeUpdate
IBeforeUpdate, 162
Simulation, 720

Behaviour, 88
AddBehaviour< T >, 89
DestroyBehaviour, 89
GetBehaviour< T >, 89
NetworkBehaviourBufferInterpolator, 317
NetworkBehaviourId, 321
RpcHeader, 689
TryGetBehaviour< T >, 89

BehaviourCount
NetworkObjectHeader, 435

BehaviourMeta
NetworkProjectConfigAsset, 500

Bits
NetworkButtons, 353

BLOCK_SIZE
NetworkId, 380

BlockByteSize
Allocator.Config, 75

BlockCount
Allocator.Config, 74

BlockShift
Allocator.Config, 74

BlockWordCount
Allocator.Config, 75

Bool
NetworkBehaviourBufferInterpolator, 309, 310

Bounds
BVHNodeDrawInfo, 221

Box
Fusion, 42

Box2D
Fusion.LagCompensation, 56

BoxExtents
ColliderDrawInfo, 222
Hitbox, 130

BoxOverlapQuery, 216
BoxOverlapQuery, 216, 217
Center, 217
Check, 217
Extents, 218
Rotation, 218

BoxOverlapQueryParams, 218
BoxOverlapQueryParams, 219
Center, 219
Extents, 219
QueryParams, 219
Rotation, 220
StaticHitsCapacity, 220

BroadRadius
HitboxRoot, 149

BUCKET_COUNT
Allocator, 70

BUCKET_INVALID
Allocator, 70

Buffer
 SimulationInput.Buffer, 748

BuildType
 NetworkRunner, 562

BuildTypes
 NetworkProjectConfig, 494
 NetworkRunner, 518

BVHDepth
 HitboxManager, 145

BVHDraw, 220
 GetEnumerator, 220
 LagCompensationDraw, 226

BVHNodeDrawInfo, 221
 Bounds, 221
 Depth, 221
 MaxDepth, 221

BVHNodes
 HitboxManager, 145

ByteCount
 NetworkObjectHeader, 437

Bytes
 Fusion, 52

CachedStaticCollidersSize
 LagCompensationSettings, 245

CanAllocateUserPayload
 SimulationMessage, 756

CanReceiveRenderCallback
 SimulationBehaviour, 739

CanReceiveSimulationCallback
 SimulationBehaviour, 739

CanSpawn
 NetworkRunner, 562

Capacity
 FixedBufferPropertyAttribute, 120
 NetworkDictionary< K, V >, 364
 NetworkDictionaryReadOnly< K, V >, 368
 NetworkLinkedList< T >, 394
 NetworkLinkedListReadOnly< T >, 400
 NetworkString< TSize >, 628
 SimulationMessage, 761

CapacityAttribute, 90
 CapacityAttribute, 90
 Length, 91

Capsule
 Fusion, 42

CapsuleBottomCenter
 ColliderDrawInfo, 222

CapsuleExtents
 ColliderDrawInfo, 222
 Hitbox, 130

CapsuleRadius
 Hitbox, 130

CapsuleTopCenter
 ColliderDrawInfo, 223

CELL_SIZE
 Simulation.AreaOfInterest, 737

Center
 AABB, 215

BoxOverlapQuery, 217

BoxOverlapQueryParams, 219

SphereOverlapQuery, 242

SphereOverlapQueryParams, 244

Changed
 NetworkBehaviour.ChangeDetector.Enumerable, 294

ChangedTick
 NetworkBehaviour, 287

Channel
 RpcAttribute, 685
 RpcInfo, 692

Check
 BoxOverlapQuery, 217
 Query, 231
 RaycastQuery, 237
 SphereOverlapQuery, 242

CheckNetworkedPropertiesBeingEmpty
 NetworkProjectConfig, 494

CheckRpcAttributeUsage
 NetworkProjectConfig, 494

Clamp
 Angle, 77

ClampSelection
 TickRate, 812

Clear
 FixedSize< T >, 112
 NetworkArray< T >, 257
 NetworkDictionary< K, V >, 361
 NetworkLinkedList< T >, 391
 NetworkPrefabTable, 482
 SerializableDictionary< TKey, TValue >, 709
 SimulationInput, 746
 SimulationInput.Buffer, 749

ClearMonitoredNetworkObjects
 NetworkObjectStatisticsManager, 796

ClearPlayerAreaOfInterest
 NetworkRunner, 520

Client
 Fusion, 42, 51
 TickRate, 816
 TickRate.Resolved, 818
 TickRate.Selection, 820

ClientSend
 TickRate.Resolved, 818

ClientSendDelta
 TickRate.Resolved, 819

ClientSendIndex
 TickRate.Selection, 820

ClientSendIndexOutOfRange
 TickRate, 811

ClientServer
 Fusion, 50, 52

ClientTickDelta
 TickRate.Resolved, 819

ClientTickStride
 TickRate.Resolved, 819

ClientToClientWithServerProxy

NetworkConfiguration, 354
ClientToServer
 NetworkConfiguration, 354
Clone
 NetworkSimulationConfiguration, 596
 SimulationMessage, 756
CloneArray< T >
 NetworkBehaviourUtils, 324
CollectGarbage
 DynamicHeap, 97
CollectGarbageDelegate
 DynamicHeap, 98
Collider
 LagCompensatedHit, 212
Collider2D
 LagCompensatedHit, 212
ColliderDrawInfo, 222
 BoxExtents, 222
 CapsuleBottomCenter, 222
 CapsuleExtents, 222
 CapsuleTopCenter, 223
 LocalToWorldMatrix, 223
 Offset, 223
 Radius, 223
 Type, 223
ColliderIndex
 Hitbox, 132
Compare
 NetworkObjectSortKeyComparer, 448
 NetworkString< TSize >, 606, 607
 Tick.RelationalComparer, 806
Compare< TOtherSize >
 NetworkString< TSize >, 607, 608
CompareOperator
 Fusion, 40
Comparer
 NetworkId, 381
 NetworkObjectTypeid, 457
 PlayerRef, 647
CompareTo
 NetworkId, 377
 NetworkObjectGuid, 425
 NetworkPrefabId, 466
 NetworkPrefabRef, 474
 Tick, 800
Completed
 IAsyncOperation, 157
CompleteSnapshot
 FusionStatisticsManager, 791
Compress
 FloatUtils, 126
ComputeHash
 IDataEncryption, 106
Config
 Allocator.Config, 72
 HitboxRoot, 149
 NetworkProjectConfigAsset, 500
 NetworkRunner, 563
 Simulation, 727
 StartGameArgs, 784
ConfigFlags
 HitboxRoot, 147
ConnectAttempts
 NetConfig, 778
 NetworkConfiguration, 355
ConnectInterval
 NetConfig, 778
 NetworkConfiguration, 355
ConnectionDefaultRtt
 NetConfig, 778
 NetworkConfiguration, 356
ConnectionGroups
 NetConfig, 778
ConnectionPingInterval
 NetConfig, 779
 NetworkConfiguration, 356
ConnectionRefused
 Fusion, 51
ConnectionSendBuffers
 NetConfig, 779
ConnectionShutdownTime
 NetConfig, 779
 NetworkConfiguration, 355
ConnectionsPerGroup
 NetConfig, 781
ConnectionTimeout
 Fusion, 51
 NetConfig, 779
 NetworkConfiguration, 355
ConnectionToken
 StartGameArgs, 784
ConnectionType
 Fusion, 41
ConsumeTick
 TickAccumulator, 808
Contains
 NetworkLinkedList< T >, 392
 NetworkLinkedListReadOnly< T >, 399
 NetworkPrefabTable, 483
 NetworkString< TSize >, 608, 609
 SimulationInput.Buffer, 749
Contains< TOtherSize >
 NetworkString< TSize >, 610
ContainsKey
 NetworkDictionary< K, V >, 361
 SerializableDictionary< TKey, TValue >, 709
ContainsValue
 NetworkDictionary< K, V >, 361
CounterMask
 Fusion, 44
ContinueWhenAll
 TaskManager, 85
Convert< T >
 NetworkInput, 383
CopyBackingFieldsToState
 NetworkBehaviour, 270

CopyFrom
 FixedArray< T >, 113
 NetworkArray< T >, 258
 SimulationInput, 746

CopyFromNetworkArray< T >
 NetworkBehaviourUtils, 324

CopyFromNetworkDictionary< D, K, V >
 NetworkBehaviourUtils, 325

CopyFromNetworkList< T >
 NetworkBehaviourUtils, 325

CopySortedTo
 SimulationInput.Buffer, 749

CopyStateFrom
 NetworkBehaviour, 270
 NetworkObject, 410, 411

CopyStateToBackingFields
 NetworkBehaviour, 270

CopyTo
 FixedArray< T >, 114
 NetworkArray< T >, 259

Count
 Fusion, 52
 NetworkDictionary< K, V >, 364
 NetworkDictionaryReadOnly< K, V >, 368
 NetworkLinkedList< T >, 394
 NetworkLinkedListReadOnly< T >, 400
 SerializableDictionary< TKey, TValue >, 712
 SimulationInput.Buffer, 751
 TickRate, 816

Create
 NetCommandHeader, 776
 NetworkSimulationConfiguration, 596
 RpcHeader, 687, 688

Create< T >
 FixedArray< T >, 114

Create< TActual, TAdapted >
 FixedArray< T >, 115

Create< TKey, TValue >
 SerializableDictionary< TKey, TValue >, 710

CreateFromFieldSequence< T >
 FixedArray< T >, 115

CreateFromIpPort
 NetAddress, 769

CreateFromSeconds
 TickTimer, 822

CreateFromTicks
 TickTimer, 822

Current
 FixedArray< T >.Enumerator, 119
 NetworkArray< T >.Enumerator, 263
 NetworkBehaviour.ChangeDetector.Enumerator, 296
 NetworkDictionary< K, V >.Enumerator, 366
 NetworkLinkedList< T >.Enumerator, 396

CurrentConnectionType
 NetworkRunner, 563

CurrentTypeId
 NetworkProjectConfig, 494

CurrentVersion
 NetworkProjectConfig, 494

Custom
 Fusion, 45, 50

CustomAuthenticationFailed
 Fusion, 51

CustomCallbackInterfaces
 StartGameArgs, 784

CustomLobbyName
 StartGameArgs, 785

CustomPhotonAppSettings
 StartGameArgs, 785

CustomPublicAddress
 StartGameArgs, 785

CustomSTUNServer
 StartGameArgs, 785

Data
 _128, 61
 _16, 62
 _2, 63
 _256, 64
 _32, 65
 _4, 66
 _512, 67
 _64, 67
 _8, 68
 NetworkInput, 385
 NetworkPrefabAcquireContext, 463
 NetworkPrefabInfo, 471
 NetworkTRSP, 635
 SimulationInput, 747

DataConsistency
 SimulationConfig, 743

DataEncryptor, 104
 EncryptData, 105

DataProperty
 IUnityValueSurrogate< T >, 192
 UnityArraySurrogate< T, ReaderWriter >, 194
 UnityDictionarySurrogate< TKeyType, TKeyReaderWriter, TValueType, TValueReaderWriter >, 196
 UnityLinkedListSurrogate< T, ReaderWriter >, 198
 UnityValueSurrogate< T, TReaderWriter >, 202

Debug
 NetworkRunner, 518

Decompress
 FloatUtils, 126

DecryptData
 IDataEncryption, 106

Default
 HitboxRoot, 147
 NetworkedAttribute, 369
 NetworkPrefabTableOptions, 489

DEFAULT_ACCURACY
 FloatUtils, 126

DEFAULT_BLOCK_COUNT
 Allocator.Config, 74

DEFAULT_BLOCK_SHIFT

Allocator.Config, 74
DefaultForPropertyAttribute, 91
 DefaultForPropertyAttribute, 91
 PropertyName, 92
 WordCount, 92
 WordOffset, 92
DefaultMtu
 NetConfig, 779
DefaultResourceName
 NetworkProjectConfig, 495
Defaults
 NetConfig, 781
Degrees
 Fusion, 52
DegreesPerSecond
 Fusion, 52
DelayMax
 NetworkSimulationConfiguration, 597
DelayMin
 NetworkSimulationConfiguration, 597
DelayPeriod
 NetworkSimulationConfiguration, 597
DelayShape
 NetworkSimulationConfiguration, 597
DelayThreshold
 NetworkSimulationConfiguration, 597
Delegate
 RpcInvokeData, 694
DeltaTime
 NetworkRunner, 563
 Simulation, 727
Depth
 BVHNodeDrawInfo, 221
Description
 INetworkAssetSource< T >, 172
Deserialize
 NetworkProjectConfig, 492
Despawn
 NetworkRunner, 520
Despawned
 IDespawned, 163
 NetworkBehaviour, 270
DestroyBehaviour
 Behaviour, 89
DestroySingleton< T >
 NetworkRunner, 521
DestroyWhenStateAuthorityLeaves
 Fusion, 43
DetectChanges
 NetworkBehaviour.ChangeDetector, 292, 293
Direct
 Fusion, 41
Direction
 RaycastQuery, 238
 RaycastQueryParams, 239
DisableNATPunchthrough
 StartGameArgs, 785
DisableSharedModelInterpolation
 NetworkTransform, 632
Disconnect
 NetworkRunner, 521
DisconnectedByPluginLogic
 Fusion, 50
DisconnectReason
 Fusion.Protocol, 57
DisplayAsEnumAttribute, 92
Dispose
 FixedArray< T >.Enumerator, 118
 NetworkArray< T >.Enumerator, 263
 NetworkDictionary< K, V >.Enumerator, 365
 NetworkLinkedList< T >.Enumerator, 395
 SimulationBehaviourListScope, 742
Distance
 LagCompensatedHit, 212
DofAttributeBase, 93
DontDestroyOnLoad
 Fusion, 43, 45
 NetworkPrefabAcquireContext, 463
DrawGizmos
 Hitbox, 129
 HitboxRoot, 148
DrawIfAttribute, 93
 DrawIfAttribute, 94, 95
 Mode, 95
DrawInfo
 HitboxManager, 145
DynamicHeap, 95
 _debruijnTable, 99
 CollectGarbage, 97
 CollectGarbageDelegate, 98
 Free, 98
 SetForcedAlive< T >, 98
DynamicHeap.Ignore, 99
DynamicHeapInstance, 99
 Allocate, 100
 AllocateArray< T >, 100
 AllocateArrayPointers< T >, 101
 AllocateTracked< T >, 101
 AllocateTrackedArray< T >, 102
 AllocateTrackedArrayPointers< T >, 102
 DynamicHeapInstance, 100
 Free, 104
DynamicWordCount
 NetworkBehaviour, 288
 NetworkMecanimAnimator, 407
ELEMENT_WORDS
 NetworkLinkedList< T >, 394
 NetworkLinkedListReadOnly< T >, 400
Empty
 NetworkObjectGuid, 429
 NetworkPrefabRef, 479
EnableAutoUpdate
 HostMigrationConfig, 151
EnableClientSessionCreation
 StartGameArgs, 785
Enabled

LagCompensationSettings, 245
 NetworkSimulationConfiguration, 598
EncryptData
 DataEncryptor, 105
 IDataEncryption, 107
EncryptionConfig
 NetworkProjectConfig, 495
EndsWith
 NetworkString< TSize >, 611
EndsWith< TOtherSize >
 NetworkString< TSize >, 611
EnqueueIncompleteSynchronousSpawns
 NetworkProjectConfig, 495
EnsureRootComponentExists< T, TStopOn >
 NestedComponentUtilities, 249
EnsureRunnerScenesActive
 NetworkRunner, 521
EntryKeyPropertyPath
 SerializableDictionary< TKey, TValue >, 712
Enumerator
 FixedArray< T >.Enumerator, 118
 NetworkArray< T >.Enumerator, 262
Equal
 Fusion, 41
Equals
 Allocator.Config, 72, 73
 Angle, 77, 78
 FloatCompressed, 122, 123
 NetworkBehaviourId, 319
 NetworkBool, 341
 NetworkButtons, 344, 345
 NetworkId, 378
 NetworkId.EqualityComparer, 382
 NetworkLoadSceneParameters, 402
 NetworkObjectGuid, 425
 NetworkObjectGuid.EqualityComparer, 430
 NetworkObjectHeader, 432
 NetworkObjectNestingKey, 442
 NetworkObjectNestingKey.EqualityComparer, 444
 NetworkObjectTypeId, 452, 453
 NetworkObjectTypeId.EqualityComparer, 459
 NetworkPrefabId, 466
 NetworkPrefabId.EqualityComparer, 470
 NetworkPrefabRef, 475
 NetworkPrefabRef.EqualityComparer, 480
 NetworkRunnerUpdaterDefaultInvokeSettings, 575, 576
 NetworkSceneInfo, 586
 NetworkSceneLoadId, 590, 591
 NetworkSceneObjectId, 592, 593
 NetworkString< TSize >, 612, 613
 PlayerRef, 643
 Ptr, 651, 652
 Ptr.EqualityComparer, 656
 QuaternionCompressed, 658
 SceneRef, 701
 Tick, 800
 Tick.EqualityComparer, 805
 Vector2Compressed, 837
 Vector3Compressed, 841
 Vector4Compressed, 847
Equals< TOtherSize >
 NetworkString< TSize >, 613, 614
Error
 Fusion, 50
 Fusion.Protocol, 57
 IAsyncOperation, 157
 NetworkSceneAsyncOp, 582
 TickRate, 811
ErrorMessage
 StartGameResult, 790
Eventual
 SimulationConfig, 743
ExecutionOrder
 NetworkProjectConfigAsset.SerializableSimulationBehaviourMeta, 501
Exists
 NetworkRunner, 522
ExpansionFactor
 LagCompensationSettings, 246
Expired
 TickTimer, 823
ExpiredOrNotRunning
 TickTimer, 823
Extents
 AABB, 215
 BoxOverlapQuery, 218
 BoxOverlapQueryParams, 219
Failed
 Fusion, 43
FailedClientCantSpawn
 Fusion, 45
FailedLocalPlayerNotYetSet
 Fusion, 45
FailedToCreateInstance
 Fusion, 45
FailedToLoadPrefabSynchronously
 Fusion, 45
FieldsMask
 FieldsMask< T >, 109, 110
FieldsMask< T >, 108
 FieldsMask, 109, 110
 operator Mask256, 110
FindObject
 NetworkRunner, 522
FindObjectsOfTypeInOrder< T >
 NestedComponentUtilities, 249
FindObjectsOfTypeInOrder< T, TCast >
 NestedComponentUtilities, 251
FirstChild
 Fusion, 53
FixedArray
 FixedArray< T >, 112
FixedArray< T >, 111
 Clear, 112
 CopyFrom, 113

CopyTo, 114
Create< T >, 114
Create< TActual, TAdapted >, 115
CreateFromFieldSequence< T >, 115
FixedArray, 112
GetEnumerator, 116
IndexOf< T >, 116
Length, 117
this[int index], 117
ToArray, 116
ToListString, 116
ToString, 117
FixedArray< T >.Enumerator, 117
Current, 119
Dispose, 118
Enumerator, 118
MoveNext, 118
Reset, 119
FixedBufferPropertyAttribute, 119
Capacity, 120
FixedBufferPropertyAttribute, 120
SurrogateType, 120
Type, 120
FixedStorage, 121
GetWordCount< T >, 121
FixedUpdateNetwork
 NetworkBehaviour, 271
 SimulationBehaviour, 739
FLAG_ADDRESSABLE
 SceneRef, 705
FLAG_DUMMY
 SimulationMessage, 761
FLAG_INTERNAL
 SimulationMessage, 761
FLAG_NOT_TICK_ALIGNED
 SimulationMessage, 762
FLAG_REMOTE
 SimulationMessage, 762
FLAG_STATIC
 SimulationMessage, 762
FLAG_TARGET_PLAYER
 SimulationMessage, 762
FLAG_TARGET_SERVER
 SimulationMessage, 762
FLAG_UNRELIABLE
 SimulationMessage, 762
FLAG_USER_FLAGS_START
 SimulationMessage, 763
FLAG_USER_MESSAGE
 SimulationMessage, 763
Flags
 NetworkObject, 415
 NetworkObjectHeader, 435
 SimulationMessage, 763
FLAGS_RESERVED
 SimulationMessage, 763
FLAGS_RESERVED_BITS
 SimulationMessage, 763
Float
 NetworkBehaviourBufferInterpolator, 310
FloatCompressed, 122
 Equals, 122, 123
 GetHashCode, 123
 operator float, 123
 operator FloatCompressed, 124
 operator!=, 124
 operator==, 124
 valueEncoded, 125
FloatUtils, 125
 Compress, 126
 Decompress, 126
 DEFAULT_ACCURACY, 126
Forward
 Fusion, 51
ForwardTicks
 FusionStatisticsSnapshot, 792
Frames
 Fusion, 52
FramesPerSecond
 Fusion, 52
Free
 DynamicHeap, 98
 DynamicHeapInstance, 104
From
 Fusion, 46
 NetworkBehaviourBufferInterpolator, 317
FromActorId
 NetAddress, 769
FromAsyncOperation
 NetworkSceneAsyncOp, 579
FromCompleted
 NetworkSceneAsyncOp, 580
FromCoroutine
 NetworkSceneAsyncOp, 580
FromCustom
 NetworkObjectTypeid, 453
FromEncoded
 PlayerRef, 643
FromError
 NetworkSceneAsyncOp, 581
FromIndex
 NetworkPrefabId, 466
 PlayerRef, 644
 SceneRef, 702
FromLocal
 RpcInfo, 691
FromMessage
 RpcInfo, 691
FromPath
 SceneRef, 702
FromPrefabId
 NetworkObjectTypeid, 453
FromRaw
 NetworkPrefabId, 467
 SceneRef, 703
FromSceneRefAndObjectIndex

NetworkObjectTypeId, 454
 FromStruct
 NetworkObjectTypeId, 454
 FromTask
 NetworkSceneAsyncOp, 581
 Full
 SimulationConfig, 743
 SimulationInput.Buffer, 751
 FullCone
 Fusion.Sockets.Stun, 59
 Fusion, 29
 After, 53
 All, 49
 AllowStateAuthorityOverride, 43
 AlreadyRunning, 51
 AreaOfInterest, 43
 AuthenticationTicketExpired, 51
 AutoHostOrClient, 42
 Before, 53
 Box, 42
 Bytes, 52
 Capsule, 42
 Client, 42, 51
 ClientServer, 50, 52
 CompareOperator, 40
 ConnectionRefused, 51
 ConnectionTimeout, 51
 ConnectionType, 41
 ConterMask, 44
 Count, 52
 Custom, 45, 50
 CustomAuthenticationFailed, 51
 Degrees, 52
 DegreesPerSecond, 52
 DestroyWhenStateAuthorityLeaves, 43
 Direct, 41
 DisconnectedByPluginLogic, 50
 DontDestroyOnLoad, 43, 45
 Equal, 41
 Error, 50
 Failed, 43
 FailedClientCantSpawn, 45
 FailedLocalPlayerNotYetSet, 45
 FailedToCreateInstance, 45
 FailedToLoadPrefabSynchronously, 45
 FirstChild, 53
 Forward, 51
 Frames, 52
 FramesPerSecond, 52
 From, 46
 GameClosed, 51
 GameIdAlreadyExists, 51
 GameIsFull, 51
 GameMode, 41
 GameNotFound, 51
 GlobalObjectInterest, 43
 Greater, 41
 GreaterOrEqual, 41
 HasMainNetworkTRSP, 43
 High, 46
 HitboxTypes, 42
 HitOptions, 42
 Host, 42, 51
 HostMigration, 51
 Ignore, 43
 IgnoreInputAuthority, 42
 IncludeBox2D, 42
 IncludePhysX, 42
 IncompatibleConfiguration, 50
 Initial, 44
 InProgress, 44
 InputAuthority, 49
 InsufficientSourceAuthority, 47, 48
 InsufficientTargetAuthority, 47
 InternalStruct, 45
 Interpolated, 46
 Invalid, 45, 50
 InvalidArguments, 51
 InvalidAuthentication, 51
 InvalidRegion, 51
 Invoked, 47
 IsZero, 41
 Kilobytes, 52
 LastChild, 53
 Latest, 46
 Less, 41
 LessOrEqual, 41
 LoadError, 44
 Local, 46
 Low, 46
 Lowest, 46
 MaskBroadcast, 49
 MaskCulled, 49
 MaskNotSent, 49
 MaskSent, 48
 MaskVersion, 43
 MasterClientObject, 43
 MaxCcuReached, 51
 Medium, 46
 Megabytes, 52
 MilliSecs, 52
 Multiplier, 52
 NetworkObjectAcquireResult, 42
 NetworkObjectFlags, 43
 NetworkObjectHeaderFlags, 43
 NetworkObjectSpawnDelegate, 53
 NetworkPrefabTableGetPrefabResult, 44
 NetworkSceneInfoChangeSource, 44
 NetworkSceneInfoDefaultFlags, 44
 NetworkSpawnFlags, 44
 NetworkSpawnStatus, 45
 NetworkTypeIdKind, 45
 NoActiveConnections, 48
 None, 41–44, 48, 52
 Normalized, 52
 NormalizedPercentage, 52

NotCulled, 48
NotEqual, 41
NotFound, 44
NotInvokableDuringResim, 47, 48
NotInvokableLocally, 47
NotSentBroadcastNoActiveConnections, 48
NotSentBroadcastNoConfirmedNorInterested-
Clients, 48
NotSentTargetClientNotAvailable, 48
NotSentTargetObjectNotConfirmed, 48
NotSentTargetObjectNotInPlayerInterest, 48
NotZero, 41
Ok, 50
OperationCanceled, 51
OperationTimeout, 51
Packets, 52
PageSizes, 45
PayloadSizeExceeded, 47, 48
Percentage, 52
PerSecond, 52
PhotonCloudTimeout, 51
Player, 46
Prefab, 45
PriorityLevel, 46
Proxies, 49
Queued, 45
Radians, 52
RadiansPerSecond, 52
Relayed, 41
Reliable, 47
Remote, 44, 46, 49
RenderSource, 46
RenderTimeframe, 46
Resimulate, 51
Retry, 43
RpcChannel, 47
RpcHostMode, 47
RpcInvokeDelegate, 53
RpcLocalInvokeResult, 47
RpcSendCullResult, 48
RpcSendMessageResult, 48
RpcSources, 49
RpcStaticInvokeDelegate, 53
RpcTargets, 49
RpcTargetStatus, 49
SceneCountMask, 44
SceneObject, 45
ScriptHeaderBackColor, 50
ScriptHeaderIcon, 50
Seconds, 52
Self, 49
SentBroadcast, 48
SentToServerForForwarding, 48
SentToTargetClient, 48
Server, 41, 51
ServerInRoom, 50
SessionLobby, 50
Shared, 41, 50, 52
SharedModeStateAuthLocalPlayer, 45
SharedModeStateAuthMasterClient, 45
ShutdownReason, 50
SimulationModes, 51
SimulationStages, 51
Single, 41
SourcesHostPlayer, 47
SourcesServer, 47
Spawned, 45
SpawnedByClient, 43
Sphere, 42
SquareMagnitude, 52
StateAuthority, 49
Struct, 43
StructArray, 43
SubtickAccuracy, 42
Success, 43, 44
TagetPlayerIsNotLocal, 47
TargetPlayerIsLocalButRpclsNotInvokableLocally,
48
TargetPlayerIsNotLocal, 47
TargetPlayerUnreachable, 48
Ticks, 52
TicksPerSecond, 52
To, 46
Topologies, 52
Units, 52
UnityPlayerLoopSystemAddMode, 52
Unreachable, 49
Unreliable, 47
V1, 43
Fusion.Analyzer, 54
Fusion.Async, 54
Fusion.Encryption, 54
Fusion.Internal, 54
Fusion.LagCompensation, 55
 Box2D, 56
 Hitbox, 56
 HitType, 56
 None, 56
 PhysX, 56
 PreProcessingDelegate, 56
Fusion.Protocol, 57
 DisconnectReason, 57
 Error, 57
 IncompatibleConfiguration, 57
 InvalidEventCode, 57
 InvalidJoinGameMode, 57
 InvalidJoinMsgType, 57
 ServerAlreadyInRoom, 57
 ServerLogic, 57
Fusion.Runtime, 57
Fusion.Runtime.Unity, 57
Fusion.Sockets, 57
 NetCommands, 58
 NetConnectFailedReason, 58
 NetDisconnectReason, 59
 NetPacketType, 59

ServerFull, 59
 ServerRefused, 59
 Timeout, 59
Fusion.Sockets.Stun, 59
 FullCone, 59
 Invalid, 59
 NATType, 59
 OpenInternet, 59
 Symmetric, 59
 UdpBlocked, 59
Fusion.Statistics, 60
FusionStatisticsManager, 790
 CompleteSnapshot, 791
 ObjectStatisticsManager, 791
FusionStatisticsSnapshot, 791
 ForwardTicks, 792
 GeneralAllocMemoryUsedInBytes, 793
 InBandwidth, 793
 InObjectUpdates, 793
 InPackets, 793
 InputInBandwidth, 793
 InputOutBandwidth, 793
 InputReceiveDelta, 794
 InterpolationOffset, 794
 InterpolationSpeed, 794
 ObjectsAllocMemoryUsedInBytes, 794
 OutBandwidth, 794
 OutObjectUpdates, 794
 OutPackets, 795
 Resimulations, 795
 RoundTripTime, 795
 SimulationSpeed, 795
 SimulationTimeOffset, 795
 StateReceiveDelta, 795
 TimeResets, 796

GameClosed
 Fusion, 51
GameIdAlreadyExists
 Fusion, 51
GamesFull
 Fusion, 51
GameMode
 Fusion, 41
 HostMigrationToken, 152
 NetworkRunner, 563
 StartGameArgs, 786
GameNotFound
 Fusion, 51
GameObject
 LagCompensatedHit, 213
GeneralAllocMemoryUsedInBytes
 FusionStatisticsSnapshot, 793
GenerateKey
 IDataEncryption, 107
Get
 NetworkArray< T >, 260
 NetworkDictionary< K, V >, 361
 NetworkDictionaryReadOnly< K, V >, 367

NetworkLinkedList< T >, 392
 NetworkLinkedListReadOnly< T >, 399
 NetworkString< TSize >, 614
 SimulationInput.Buffer, 750
 TickRate, 812
Get< T >
 NetworkInput, 383
GetAllBehaviours
 NetworkRunner, 522
GetAllBehaviours< T >
 NetworkRunner, 523
GetAllNetworkBehaviourTypes
 ReflectionUtils, 677
GetAllSimulationBehaviourTypes
 ReflectionUtils, 678
GetAllWeavedAssemblies
 ReflectionUtils, 678
GetAllWeavedNetworkBehaviourTypes
 ReflectionUtils, 678
GetAllWeavedSimulationBehaviourTypes
 ReflectionUtils, 678
GetAllWeaverGeneratedTypes
 ReflectionUtils, 679
GetAreaOfInterestGizmoData
 NetworkRunner, 524
 Simulation, 720
GetArrayReader< T >
 NetworkBehaviour, 271
GetAwaiter
 NetworkSceneAsyncOp, 581
 NetworkSpawnOp, 600
GetBehaviour< T >
 Behaviour, 89
GetBehaviourChangedTickArray
 NetworkObjectHeader, 432
GetBehaviourReader< T >
 NetworkBehaviour, 272, 273
GetBehaviourReader< TBehaviour, TProperty >
 NetworkBehaviour, 273
GetByteArrayHashCode
 ReadWriteUtilsForWeaver, 670
GetByteCountUtf8NoHash
 ReadWriteUtilsForWeaver, 670
GetCapacity< TSize >
 NetworkString< TSize >, 615
GetCellSize
 Simulation.AreaOfInterest, 734
GetChangeDetector
 NetworkBehaviour, 274
GetCharCount
 NetworkString< TSize >, 615
GetCurrentVersion
 Versioning, 650
GetCustomAttributeOrThrow< T >
 ReflectionUtils, 679
GetData
 SimulationMessage, 757
GetDataPointer

NetworkObjectHeader, 433
GetDataWordCount
 NetworkObjectHeader, 433
GetDictionaryReader< K, V >
 NetworkBehaviour, 274, 275
GetDivisor
 TickRate, 812
GetElementHashCode
 IElementReaderWriter< T >, 164
GetElementWordCount
 IElementReaderWriter< T >, 164
GetEntries
 NetworkPrefabTable, 483
GetEnumerator
 BVHDraw, 220
 FixedArray< T >, 116
 HitboxColliderContainerDraw, 224
 NetworkArray< T >, 260
 NetworkBehaviour.ChangeDetector.Enumerable,
 294
 NetworkDictionary< K, V >, 361
 NetworkLinkedList< T >, 392
 NetworkString< TSize >, 615
 SerializableDictionary< TKey, TValue >, 710
 SnapshotHistoryDraw, 240
GetExecutionOrder
 NetworkProjectConfig, 493
GetFlag
 SimulationMessage, 757
GetGuid
 NetworkPrefabTable, 483
GetHashCode
 Allocator.Config, 73
 Angle, 78
 FloatCompressed, 123
 NetworkBehaviourId, 319
 NetworkBool, 342
 NetworkButtons, 345
 NetworkId, 378
 NetworkId.EqualityComparer, 382
 NetworkLoadSceneParameters, 402
 NetworkObjectGuid, 426
 NetworkObjectGuid.EqualityComparer, 430
 NetworkObjectHeader, 433
 NetworkObjectNestingKey, 442
 NetworkObjectNestingKey.EqualityComparer, 445
 NetworkObjectTypeId, 455
 NetworkObjectTypeId.EqualityComparer, 460
 NetworkPrefabId, 467
 NetworkPrefabId.EqualityComparer, 470
 NetworkPrefabRef, 475
 NetworkPrefabRef.EqualityComparer, 480
 NetworkRunnerUpdaterDefaultInvokeSettings, 576
 NetworkSceneInfo, 587
 NetworkSceneLoadId, 591
 NetworkSceneObjectId, 593
 NetworkString< TSize >, 615
 PlayerRef, 644
 Ptr, 652
 Ptr.EqualityComparer, 657
 QuaternionCompressed, 659
 SceneRef, 703
 Tick, 801
 Tick.EqualityComparer, 805
 Vector2Compressed, 837
 Vector3Compressed, 842
 Vector4Compressed, 847
GetId
 NetworkPrefabTable, 483
GetInput< T >
 NetworkBehaviour, 275
GetInputAuthority
 Simulation, 721
GetInputForPlayer
 Simulation, 721
GetInputForPlayer< T >
 NetworkRunner, 524
GetInsertTime
 SimulationInput.Buffer, 750
GetInstanceCount
 NetworkPrefabTable, 484
GetInstanceEnumerator
 NetworkRunner, 524
GetInterfaceListHead
 NetworkRunner, 524
GetInterfaceListNext
 NetworkRunner, 525
GetInterfaceListPrev
 NetworkRunner, 525
GetInterfaceListsCount
 NetworkRunner, 525
GetLastUsedInputHeader
 SimulationInput.Buffer, 751
GetLinkListReader< T >
 NetworkBehaviour, 276
GetLocalAuthorityMask
 NetworkBehaviour, 277
 NetworkObject, 411
GetMainNetworkTRSPData
 NetworkObjectHeader, 434
GetMaxWordCount
 NetworkInputUtils, 386
GetMetaData
 NetworkBehaviourUtils, 326
GetNestedComponentInChildren< T, TStopOn >
 NestedComponentUtilities, 252
GetNestedComponentInParent< T, TStopOn >
 NestedComponentUtilities, 253
GetNestedComponentInParents< T, TStopOn >
 NestedComponentUtilities, 253
GetNestedComponentsInChildren< T >
 NestedComponentUtilities, 253
GetNestedComponentsInChildren< T, TSearch, TStop
 >
 NestedComponentUtilities, 254
GetNestedComponentsInChildren< T, TStopOn >

NestedComponentUtilities, 254
 GetNestedComponentsInParents< T >
 NestedComponentUtilities, 254
 GetNestedComponentsInParents< T, TStop >
 NestedComponentUtilities, 255
 GetNetworkObjectStatistics
 NetworkObjectStatisticsManager, 796
 GetObjectsAndPlayersInAreaOfInterestCell
 Simulation, 721
 GetParentComponent< T >
 NestedComponentUtilities, 255
 GetPhysicsScene
 NetworkRunner, 526
 GetPhysicsScene2D
 NetworkRunner, 526
 GetPlayerActorId
 NetworkRunner, 526
 GetPlayerConnectionToken
 NetworkRunner, 527
 GetPlayerConnectionType
 NetworkRunner, 527
 GetPlayerObject
 NetworkRunner, 527
 GetPlayerRtt
 NetworkRunner, 528
 GetPlayerTickAndAlpha
 HitboxManager, 134
 GetPlayerUserId
 NetworkRunner, 528
 GetPressed
 NetworkButtons, 345
 GetProductVersion
 Versioning, 650
 GetPropertyReader< T >
 NetworkBehaviour, 277
 GetPropertyReader< TBehaviour, TProperty >
 NetworkBehaviour, 278
 GetRawInputForPlayer
 NetworkRunner, 528
 GetRef< T >
 NetworkArrayExtensions, 264
 GetReleased
 NetworkButtons, 346
 GetRenderBuffers
 RenderTimeline, 682
 GetResult
 NetworkSceneAsyncOp.Awaiter, 584
 NetworkSpawnOp.Awaiter, 602
 GetResumeSnapshotNetworkObjects
 NetworkRunner, 528
 GetResumeSnapshotNetworkSceneObjects
 NetworkRunner, 529
 GetRpcStaticIndexOrThrow
 NetworkBehaviourUtils, 326
 GetRpcTargetStatus
 NetworkRunner, 529
 GetRunnerForObject
 NetworkRunner, 529
 GetRunnerForScene
 NetworkRunner, 530
 GetSceneRef
 INetworkSceneManager, 185, 186
 GetSingleton< T >
 NetworkRunner, 530
 GetSource
 NetworkPrefabTable, 484, 485
 GetStateAuthority
 Simulation, 721
 GetStaticWordCount
 NetworkBehaviourUtils, 327
 GetStringHashCode
 ReadWriteUtilsForWeaver, 671
 GetTickRate
 TickRate, 813
 GetType
 NetworkInputUtils, 386
 GetTypeKey
 NetworkInputUtils, 387
 GetVersion
 NetworkObjectFlagsExtensions, 419
 GetWeavedAttributeOrThrow
 ReflectionUtils, 680
 GetWordCount
 NetworkBehaviourUtils, 327
 NetworkInputUtils, 387
 NetworkObject, 411
 GetWordCount< T >
 FixedStorage, 121
 NetworkStructUtils, 629
 GetWordCountString
 ReadWriteUtilsForWeaver, 671
 GizmosColor
 Hitbox, 131
 HitboxRoot, 150
 GizmosDrawWireCapsule
 LagCompensationDraw, 226
 Global
 NetworkProjectConfig, 498
 NetworkProjectConfigAsset, 500
 GlobalObjectInterest
 Fusion, 43
 GlobalsSize
 Allocator.Config, 74
 HeapConfiguration, 128
 Greater
 Fusion, 41
 GreaterOrEqual
 Fusion, 41
 Guid
 NetworkObjectPrefabData, 445
 HasAddress
 NetAddress, 771
 HasAnyActiveConnections
 Simulation, 722
 HasHeader
 NetworkPrefabAcquireContext, 464

NetworkPrefabInfo, 471
HasInputAuthority
 NetworkBehaviour, 288
 NetworkObject, 416
HasMainNetworkTRSP
 Fusion, 43
 NetworkObjectHeader, 434
HasSingleton< T >
 NetworkRunner, 530
HasStateAuthority
 NetworkBehaviour, 288
 NetworkObject, 416
HasStaticWordCount
 NetworkBehaviourUtils, 327
Header
 NetworkPrefabInfo, 471
 SimulationInput, 747
Heap
 NetworkProjectConfig, 495
HEAP_ALIGNMENT
 Allocator, 70
HeapConfiguration, 127
 GlobalsSize, 128
 Init, 127
 PageCount, 128
 PageShift, 128
 ToString, 127
HeapSizeAllocated
 Allocator.Config, 75
HeapSizeUsable
 Allocator.Config, 75
HideNetworkObjectInactivityGuard
 NetworkProjectConfig, 495
High
 Fusion, 46
Hitbox, 128
 BoxExtents, 130
 CapsuleExtents, 130
 CapsuleRadius, 130
 ColliderIndex, 132
 DrawGizmos, 129
 Fusion.LagCompensation, 56
 GizmosColor, 131
 HitboxActive, 132
 HitboxIndex, 132
 LagCompensatedHit, 213
 Offset, 131
 OnDrawGizmos, 130
 Position, 132
 PositionRotationQueryParams, 228
 Root, 131
 SetLayer, 130
 SphereRadius, 131
 Type, 131
HitboxActive
 Hitbox, 132
HitboxBufferLengthInMs
 LagCompensationSettings, 245
 HitboxColliderContainerDraw, 224
 GetEnumerator, 224
 HitboxColliderPosition
 LagCompensatedHit, 213
 HitboxColliderRotation
 LagCompensatedHit, 213
 HitboxDefaultCapacity
 LagCompensationSettings, 245
 Hitboxes
 HitboxRoot, 150
 HitboxIndex
 Hitbox, 132
 HitboxManager, 132
 BVHDepth, 145
 BVHNodes, 145
 DrawInfo, 145
 GetPlayerTickAndAlpha, 134
 OverlapBox, 135, 136
 OverlapSphere, 137–139
 PositionRotation, 139, 140
 Raycast, 140–142
 RaycastAll, 143, 144
 TotalHitboxes, 145
 HitboxRoot, 146
 BroadRadius, 149
 Config, 149
 ConfigFlags, 147
 Default, 147
 DrawGizmos, 148
 GizmosColor, 150
 Hitboxes, 150
 HitboxRootActive, 150
 IncludeInactiveHitboxes, 147
 InInterest, 151
 InitHitboxes, 148
 IsHitboxActive, 148
 Legacy, 147
 Manager, 151
 MAX_HITBOXES, 150
 Offset, 150
 OnDrawGizmos, 148
 ReinitializeHitboxesBeforeRegistration, 147
 SetHitboxActive, 149
 SetMinBoundingRadius, 149
 HitboxRootActive
 HitboxRoot, 150
 HitboxTypes
 Fusion, 42
 HitOptions
 Fusion, 42
 HitType
 Fusion.LagCompensation, 56
 Host
 Fusion, 42, 51
 HostMigration
 Fusion, 51
 NetworkProjectConfig, 495
 SimulationConfig, 743

HostMigrationConfig, 151
 EnableAutoUpdate, 151
 UpdateDelay, 152
HostMigrationResume
 StartGameArgs, 786
HostMigrationToken, 152
 GameMode, 152
 StartGameArgs, 786
HostMode
 RpcAttribute, 685
HostPlayer
 SimulationRuntimeConfig, 766

IAfterAllTicks, 152
 AfterAllTicks, 153
IAfterClientPredictionReset, 153
 AfterClientPredictionReset, 154
IAfterHostMigration, 154
 AfterHostMigration, 154
IAfterRender, 154
 AfterRender, 155
IAfterSpawned, 155
 AfterSpawned, 155
IAfterTick, 155
 AfterTick, 156
IAfterUpdate, 156
 AfterUpdate, 156
IAsyncOperation, 157
 Completed, 157
 Error, 157
 IsDone, 157
IBeforeAllTicks, 158
 BeforeAllTicks, 158
IBeforeClientPredictionReset, 159
 BeforeClientPredictionReset, 159
IBeforeCopyPreviousState, 159
 BeforeCopyPreviousState, 160
IBeforeHitboxRegistration, 160
 BeforeHitboxRegistration, 160
IBeforeSimulation, 160
 BeforeSimulation, 161
IBeforeTick, 161
 BeforeTick, 162
IBeforeUpdate, 162
 BeforeUpdate, 162
ICoroutine, 162
Id
 NetworkBehaviour, 288
 NetworkObject, 416
 NetworkObjectHeader, 435
 NetworkObjectHeaderPtr, 438
 NetworkObjectMeta, 440
IDataEncryption, 105
 ComputeHash, 106
 DecryptData, 106
 EncryptData, 107
 GenerateKey, 107
 Setup, 108
 VerifyHash, 108

IDespawned, 163
 Despawned, 163
IElementReaderWriter< T >, 163
 GetElementHashCode, 164
 GetElementWordCount, 164
 Read, 165
 ReadRef, 165
 Write, 165
IFixedStorage, 167
Ignore
 Fusion, 43
IgnoreInputAuthority
 Fusion, 42
IInputAuthorityGained, 167
 InputAuthorityGained, 167
IInputAuthorityLost, 168
 InputAuthorityLost, 168
IInterestEnter, 168
 InterestEnter, 168
IInterestExit, 169
 InterestExit, 169
ILocalPrefabCreated, 169
 LocalPrefabCreated, 170
IMessage, 649
InBandwidth
 FusionStatisticsSnapshot, 793
 NetworkObjectStatisticsSnapshot, 798
IncludeBox2D
 Fusion, 42
IncludeInactiveHitboxes
 HitboxRoot, 147
IncludePhysX
 Fusion, 42
IncompatibleConfiguration
 Fusion, 50
 Fusion.Protocol, 57
IndexOf
 NetworkLinkedList< T >, 392
 NetworkLinkedListReadOnly< T >, 399
 NetworkSceneInfo, 587
 NetworkString< TSize >, 616–618
IndexOf< T >
 FixedArray< T >, 116
 NetworkArrayExtensions, 264
IndexOf< TOtherSize >
 NetworkString< TSize >, 619–621
INetworkArray, 170
 this[int index], 170
INetworkAssetSource< T >, 171
 Acquire, 171
 Description, 172
 IsCompleted, 172
 Release, 172
 WaitForResult, 172
INetworkDictionary, 172
 Add, 173
INetworkInput, 173
INetworkLinkedList, 173

Add, 174
INetworkObjectInitializer, 174
 InitializeNetworkState, 174
INetworkObjectProvider, 175
 AcquirePrefabInstance, 175
 ReleaseInstance, 176
INetworkPrefabSource, 176
 AssetGuid, 177
INetworkRunnerCallbacks, 177
 OnConnectedToServer, 178
 OnConnectFailed, 178
 OnConnectRequest, 178
 OnCustomAuthenticationResponse, 179
 OnDisconnectedFromServer, 179
 OnHostMigration, 179
 OnInput, 179
 OnInputMissing, 180
 OnObjectEnterAOI, 180
 OnObjectExitAOI, 180
 OnPlayerJoined, 181
 OnPlayerLeft, 181
 OnReliableDataProgress, 181
 OnReliableDataReceived, 182
 OnSceneLoadDone, 182
 OnSceneLoadStart, 182
 OnSessionListUpdated, 182
 OnShutdown, 183
 OnUserSimulationMessage, 183
INetworkRunnerUpdater, 183
 Initialize, 184
 Shutdown, 184
INetworkSceneManager, 185
 GetSceneRef, 185, 186
 Initialize, 186
 IsBusy, 188
 IsRunnerScene, 186
 LoadScene, 186
 MainRunnerScene, 188
 MakeDontDestroyOnLoad, 186
 MoveGameObjectToScene, 187
 OnSceneInfoChanged, 187
 Shutdown, 187
 TryGetPhysicsScene2D, 187
 TryGetPhysicsScene3D, 188
 UnloadScene, 188
INetworkStruct, 189
INetworkTRSPTeleport, 189
 Teleport, 189
InInterest
 HitboxRoot, 151
Init
 HeapConfiguration, 127
 NetworkBehaviour.ChangeDetector, 293
 NetworkConfiguration, 354
 TickRate, 813
 UnityArraySurrogate< T, ReaderWriter >, 193
 UnityDictionarySurrogate< TKeyType, TKeyReaderWriter, TValueType, TValueReaderWriter >, 195
 UnityLinkedListSurrogate< T, ReaderWriter >, 197
 UnitySurrogateBase, 199
 UnityValueSurrogate< T, TReaderWriter >, 201
InitHitboxes
 HitboxRoot, 148
Initial
 Fusion, 44
Initialize
 INetworkRunnerUpdater, 184
 INetworkSceneManager, 186
InitializeNetworkArray< T >
 NetworkBehaviourUtils, 328
InitializeNetworkDictionary< D, K, V >
 NetworkBehaviourUtils, 328
InitializeNetworkList< T >
 NetworkBehaviourUtils, 329
InitializeNetworkState
 INetworkObjectInitializer, 174
 NetworkObjectInitializerUnity, 439
InObjectUpdates
 FusionStatisticsSnapshot, 793
InPackets
 FusionStatisticsSnapshot, 793
 NetworkObjectStatisticsSnapshot, 798
InProgress
 Fusion, 44
INPUT
 AuthorityMasks, 87
InputAuthority
 Fusion, 49
 NetworkObject, 416
 NetworkObjectHeader, 436
 NetworkObjectMeta, 440
InputAuthorityGained
 IInputAuthorityGained, 167
InputAuthorityLost
 IInputAuthorityLost, 168
InputCount
 Simulation, 727
InputDataWordCount
 SimulationConfig, 743
InputInBandwidth
 FusionStatisticsSnapshot, 793
InputOutBandwidth
 FusionStatisticsSnapshot, 793
InputReceiveDelta
 FusionStatisticsSnapshot, 794
InputTotalWordCount
 SimulationConfig, 745
InputTransferMode
 SimulationConfig, 744
InputTransferModes
 SimulationConfig, 743
Instance
 NetworkObjectSortKeyComparer, 449
Instances
 NetworkRunner, 563

InstantiateInRunnerScene
 NetworkRunner, 530, 531

InstantiateInRunnerScene< T >
 NetworkRunner, 531

InsufficientSourceAuthority
 Fusion, 47, 48

InsufficientTargetAuthority
 Fusion, 47

Int
 NetworkBehaviourBufferInterpolator, 312

InterestEnter
 IInterestEnter, 168

InterestExit
 IInterestExit, 169

InternalOnDestroy
 NetworkBehaviourUtils, 330

InternalOnDisable
 NetworkBehaviourUtils, 330

InternalOnEnable
 NetworkBehaviourUtils, 330

InternalStruct
 Fusion, 45

InterpAlpha
 SimulationInputHeader, 752

InterpFrom
 SimulationInputHeader, 752

Interpolated
 Fusion, 46

InterpolatedErrorCorrectionSettings, 202
 MaxRate, 202
 MinRate, 203
 PosBlendEnd, 203
 PosBlendStart, 203
 PosMinCorrection, 203
 PosTeleportDistance, 204
 RotBlendEnd, 204
 RotBlendStart, 204
 RotTeleportRadians, 204

InterpolationOffset
 FusionStatisticsSnapshot, 794

InterpolationSpeed
 FusionStatisticsSnapshot, 794

InterpTo
 SimulationInputHeader, 752

Invalid
 Fusion, 45, 50
 Fusion.Sockets.Stun, 59

InvalidArguments
 Fusion, 51

InvalidAuthentication
 Fusion, 51

InvalidEventCode
 Fusion.Protocol, 57

InvalidJoinGameMode
 Fusion.Protocol, 57

InvalidJoinMsgType
 Fusion.Protocol, 57

InvalidRegion

Fusion, 51

InvalidTickRate
 TickRate, 811

InvertY
 NormalizedRectAttribute, 640

Invoked
 Fusion, 47

InvokeLocal
 RpcAttribute, 686

InvokeRenderInBatchMode
 NetworkProjectConfig, 496

InvokeRpc
 NetworkBehaviourUtils, 336

InvokeSceneLoadDone
 NetworkRunner, 531

InvokeSceneLoadStart
 NetworkRunner, 532

IPlayerJoined, 205
 PlayerJoined, 205

IPlayerLeft, 205
 PlayerLeft, 206

IRemotePrefabCreated, 206
 RemotePrefabCreated, 206

Is< T >
 NetworkInput, 384

IsAcquired
 NetworkPrefabTable, 485

IsActiveOnLoad
 NetworkLoadSceneParameters, 404

IsBeingDestroyed
 NetworkObjectReleaseContext, 447

IsBusy
 INetworkSceneManager, 188

ISceneLoadDone, 207
 SceneLoadDone, 207

ISceneLoadStart, 207
 SceneLoadStart, 208

IsClient
 NetworkRunner, 563
 Simulation, 727

IsCloudReady
 NetworkRunner, 564

IsCompleted
 INetworkAssetSource< T >, 172
 NetworkSceneAsyncOp.Awaiter, 584
 NetworkSpawnOp.Awaiter, 602

IsConnectedToServer
 NetworkRunner, 564

IsCustom
 NetworkObjectTypeId, 458

IsDone
 IAsyncOperation, 157
 NetworkSceneAsyncOp, 582

IsFailed
 NetworkSpawnOp, 600

IsFirstTick
 NetworkRunner, 564
 Simulation, 728

IsForward
 NetworkRunner, 564
 Simulation, 728
IsGlobalLoaded
 NetworkProjectConfigAsset, 501
IsHitboxActive
 HitboxRoot, 148
IsIgnored
 NetworkObjectFlagsExtensions, 419
ISimulationEnter, 208
 SimulationEnter, 208
ISimulationExit, 209
 SimulationExit, 209
IsIndex
 SceneRef, 706
IsInList
 NetBitBufferList, 773
IsInputAuthority
 Simulation, 722
IsInSimulation
 NetworkObject, 417
IsInterestedIn
 NetworkRunner, 532
 Simulation, 722
IsInvokeLocal
 RpcInfo, 692
IsIPv4
 NetAddress, 771
IsIPv6
 NetAddress, 771
IsLastTick
 NetworkRunner, 564
 Simulation, 728
IsLocalPhysics2D
 NetworkLoadSceneParameters, 404
IsLocalPhysics3D
 NetworkLoadSceneParameters, 404
IsLocalPlayerFirstExecution
 Simulation, 728
IsLocalSimulationInputAuthority
 Simulation, 723
IsLocalSimulationStateAuthority
 Simulation, 723, 724
IsLocalSimulationStateOrInputSource
 Simulation, 724
IsMainTRSP
 NetworkTRSP, 636
IsMasterClient
 PlayerRef, 647
 Simulation, 728
IsNestedObject
 NetworkObjectReleaseContext, 447
IsNone
 NetworkObjectNestingKey, 443
 NetworkObjectTypeId, 458
 NetworkPrefabId, 469
 PlayerRef, 647
IsOpen

SessionInfo, 715
StartGameArgs, 786
IsPath
 SceneRef, 703
ISpawned, 209
 Spawned, 210
IsPlayer
 NetworkRunner, 564
 Simulation, 729
IsPlayerValid
 NetworkRunner, 532
IsPrefab
 NetworkObjectTypeid, 458
IsProxy
 NetworkBehaviour, 288
 NetworkObject, 417
IsQueued
 NetworkSpawnOp, 600
IsReadOnly
 SerializableDictionary< TKey, TValue >, 712
IsRealPlayer
 PlayerRef, 648
IsRelayAddr
 NetAddress, 771
IsReserved
 NetworkId, 381
IsResimulation
 NetworkRunner, 565
 Simulation, 729
IsResume
 NetworkObject, 415
 NetworkRunner, 565
IsRunnerScene
 INetworkSceneManager, 186
IsRunning
 NetworkRunner, 565
 Simulation, 729
 TickTimer, 824
IsSceneAuthority
 NetworkRunner, 565
IsSceneManagerBusy
 NetworkRunner, 565
IsSceneObject
 NetworkObjectTypeid, 458
IsServer
 NetworkRunner, 565
 Simulation, 729
IsSet
 NetworkButtons, 346
IsSet< T >
 NetworkButtons, 346
IsSharedModeMasterClient
 NetworkRunner, 566
IsShutdown
 NetworkRunner, 566
 Simulation, 729
IsSingleLoad
 NetworkLoadSceneParameters, 404

IsSinglePlayer
 NetworkRunner, 566
 Simulation, 729

IsSpawnable
 NetworkObject, 417

IsSpawned
 NetworkSpawnOp, 600

IsStarting
 NetworkRunner, 566

IsStateAuthority
 Simulation, 724, 725

IsStruct
 NetworkObjectTypeId, 458

IsSynchronous
 NetworkPrefabAcquireContext, 463
 NetworkPrefabInfo, 471

IsTargeted
 SimulationMessage, 757

IStateAuthorityChanged, 210
 StateAuthorityChanged, 210

IsUnreliable
 SimulationMessage, 765

IsValid
 LobbyInfo, 246
 NetAddress, 771
 NetworkBehaviourId, 321
 NetworkId, 381
 NetworkInput, 385
 NetworkObject, 417
 NetworkObjectGuid, 429
 NetworkObjectNestingKey, 444
 NetworkObjectTypeId, 458
 NetworkPrefabId, 469
 NetworkPrefabRef, 479
 NetworkSceneAsyncOp, 582
 NetworkSceneObjectId, 594
 SceneRef, 706
 SessionInfo, 715
 TickRate, 813, 814

IsVersionCurrent
 NetworkObjectFlagsExtensions, 421

isVisible
 SessionInfo, 715
 StartGameArgs, 786

IsZero
 Fusion, 41

ItemsPropertyPath
 SerializableDictionary< TKey, TValue >, 712

IUnitySurrogate, 190
 Read, 190
 Write, 191

IUnityValueSurrogate< T >, 191
 DataProperty, 192

JoinSessionLobby
 NetworkRunner, 532

Key
 NetworkRpcStaticWeavedInvokerAttribute, 508

NetworkRpcWeavedInvokerAttribute, 509
 RpcInvokeData, 694

Keys
 SerializableDictionary< TKey, TValue >, 712

Kilobytes
 Fusion, 52

Kind
 NetworkObjectTypeId, 459

LagCompensatedExt, 224
 SortDistance, 224
 SortReference, 225

LagCompensatedHit, 210
 Collider, 212
 Collider2D, 212
 Distance, 212
 GameObject, 213
 Hitbox, 213
 HitboxColliderPosition, 213
 HitboxColliderRotation, 213
 Normal, 213
 operator LagCompensatedHit, 211, 212
 Point, 213
 Type, 214

LagCompensation
 NetworkProjectConfig, 496
 NetworkRunner, 566

LagCompensationDraw, 225
 BVHDraw, 226
 GizmosDrawWireCapsule, 226
 SnapshotHistoryDraw, 226

LagCompensationSettings, 244
 CachedStaticCollidersSize, 245
 Enabled, 245
 ExpansionFactor, 246
 HitboxBufferLengthInMs, 245
 HitboxDefaultCapacity, 245
 Optimize, 246

LagCompensationUtils.ContactData, 226
 Normal, 227
 Penetration, 227
 Point, 227

LastChild
 Fusion, 53

LastReceiveTick
 NetworkObject, 417

Latest
 Fusion, 46

LatestServerTick
 NetworkRunner, 566
 Simulation, 730

LatestState
 SimulationConfig, 743

LayerMask
 Query, 231
 QueryParams, 233

Legacy
 HitboxRoot, 147

Length

CapacityAttribute, 91
FixedArray< T >, 117
NetworkArray< T >, 261
NetworkArrayReadOnly< T >, 266
NetworkBehaviourBuffer, 306
NetworkString< TSize >, 628
RaycastQuery, 238
RaycastQueryParams, 239
Lerp
 Angle, 78
Less
 Fusion, 41
LessOrEqual
 Fusion, 41
Load
 NetworkPrefabTable, 485
LoadError
 Fusion, 44
LoadId
 NetworkLoadSceneParameters, 403
LoadScene
 INetworkSceneManager, 186
 NetworkRunner, 533, 534
LoadSceneMode
 NetworkLoadSceneParameters, 404
LoadSceneParameters
 NetworkLoadSceneParameters, 404
LobbyInfo, 246
 IsValid, 246
 Name, 247
 NetworkRunner, 567
 Region, 247
Local
 Fusion, 46
LocalAddress
 Simulation, 730
LocalAlpha
 NetworkRunner, 567
 Simulation, 730
LocalhostIPv4
 NetAddress, 770
LocalhostIPv6
 NetAddress, 770
LocallInvokeResult
 RpcInvokeInfo, 695
LocalPhysicsMode
 NetworkLoadSceneParameters, 405
LocalPlayer
 NetworkRunner, 567
 Simulation, 730
LocalPrefabCreated
 ILocalPrefabCreated, 170
LocalRenderTime
 NetworkRunner, 567
LocalToWorldMatrix
 ColliderDrawInfo, 223
LogSimpleUnity, 247
LossChanceMax
 NetworkSimulationConfiguration, 598
LossChanceMin
 NetworkSimulationConfiguration, 598
LossChancePeriod
 NetworkSimulationConfiguration, 598
LossChanceShape
 NetworkSimulationConfiguration, 598
LossChanceThreshold
 NetworkSimulationConfiguration, 598
Low
 Fusion, 46
Lowest
 Fusion, 46
MainRunnerScene
 INetworkSceneManager, 188
MakeDontDestroyOnLoad
 INetworkSceneManager, 186
 NetworkRunner, 534
MakeInitializer< K, V >
 NetworkBehaviour, 279
MakeInitializer< T >
 NetworkBehaviour, 279
MakePtr< T >
 NetworkBehaviour, 279
MakeRef< T >
 NetworkBehaviour, 280, 281
MakeSerializableDictionary< K, V >
 NetworkBehaviourUtils, 331
Manager
 HitboxRoot, 151
MaskBroadcast
 Fusion, 49
MaskCulled
 Fusion, 49
MaskNotSent
 Fusion, 49
MaskSent
 Fusion, 48
MaskVersion
 Fusion, 43
MASTER_CLIENT_RAW
 PlayerRef, 646
MasterClient
 PlayerRef, 648
 SimulationRuntimeConfig, 766
MasterClientObject
 Fusion, 43
MatchmakingMode
 StartGameArgs, 786
MAX
 NetworkRNG, 506
Max
 AABB, 215
 Angle, 78
MAX_HITBOXES
 HitboxRoot, 150
MAX_INDEX
 NetworkPrefabId, 468

MAX_PAYLOAD_SIZE
 SimulationMessage, 763

MAX_SCENE_OBJECT_INDEX
 NetworkObjectTypeId, 456

MaxCcuReached
 Fusion, 51

MaxConnections
 NetConfig, 779

MaxDepth
 BVHNodeDrawInfo, 221

MaxLateInputs
 TimeSyncConfiguration, 825

MaxLateSnapshots
 TimeSyncConfiguration, 826

MaxPayloadSize
 RpcAttribute, 685

MaxPlayers
 SessionInfo, 715

MaxRate
 InterpolatedErrorCorrectionSettings, 202

MaxScenes
 NetworkSceneInfo, 588

MaxSize
 NetworkSerializeMethodAttribute, 595

Medium
 Fusion, 46

Megabytes
 Fusion, 52

Message
 NetworkObjectSpawnException, 450
 SimulationMessagePtr, 765
 WarnIfAttribute, 851

MessageSize
 RpcSendResult, 696

Meta
 NetworkPrefabAcquireContext, 463

META_WORD_COUNT
 NetworkDictionary< K, V >, 364

META_WORDS
 NetworkLinkedList< T >, 394
 NetworkLinkedListReadOnly< T >, 400

Method
 RenderAttribute, 681
 RpcHeader, 690

MethodName
 OnChangedRenderAttribute, 641

MilliSecs
 Fusion, 52

Min
 AABB, 215
 Angle, 79

MinRate
 InterpolatedErrorCorrectionSettings, 203

Mode
 DrawIfAttribute, 95
 NetworkRunner, 567
 Simulation, 730

Modes

SimulationBehaviourAttribute, 741

MonitorNetworkObjectStatistics
 NetworkObjectStatisticsManager, 797

MoveGameObjectToSameScene
 NetworkRunner, 535

MoveGameObjectToScene
 INetworkSceneManager, 187
 NetworkRunner, 535

MoveNext
 FixedArray< T >.Enumerator, 118
 NetworkArray< T >.Enumerator, 263
 NetworkBehaviour.ChangeDetector.Enumerator,
 295
 NetworkDictionary< K, V >.Enumerator, 365
 NetworkLinkedList< T >.Enumerator, 396

MoveToRunnerScene
 NetworkRunner, 536

MoveToRunnerScene< T >
 NetworkRunner, 536

MtuDefault
 NetworkConfiguration, 356

Multiple
 NetworkProjectConfig, 492

Multiplier
 Fusion, 52

Name
 LobbyInfo, 247
 NetworkObject, 417
 SessionInfo, 715

NATTType
 Fusion.Sockets.Stun, 59
 NetworkRunner, 567

NestedComponentUtilities, 247
 EnsureRootComponentExists< T, TStopOn >, 249
 FindObjectsOfTypeInOrder< T >, 249
 FindObjectsOfTypeInOrder< T, TCast >, 251
 GetNestedComponentInChildren< T, TStopOn >,
 252
 GetNestedComponentInParent< T, TStopOn >,
 253
 GetNestedComponentInParents< T, TStopOn >,
 253
 GetNestedComponentsInChildren< T >, 253
 GetNestedComponentsInChildren< T, TStopOn >,
 254
 GetNestedComponentsInParents< T >, 254
 GetNestedComponentsInParents< T, TStop >,
 255
 GetParentComponent< T >, 255

NestedObjects
 NetworkObject, 415

NestingKey
 NetworkObjectHeader, 436

NestingRoot
 NetworkObjectHeader, 436

NetAddress, 767

ActorId, 770
Any, 768
AnyIPv6, 768
CreateFromIpPort, 769
FromActorId, 769
HasAddress, 771
IsIPv4, 771
IsIPv6, 771
IsRelayAddr, 771
IsValid, 771
LocalhostIPv4, 770
LocalhostIPv6, 770
NetBitBufferList, 772
 AddFirst, 772
 AddLast, 772
 IsInList, 773
 Remove, 773
 RemoveHead, 773
NetCommandAccepted, 774
NetCommandConnect, 774
NetCommandDisconnect, 775
NetCommandHeader, 775
 Create, 776
NetCommandRefused, 776
NetCommands
 Fusion.Sockets, 58
NetConfig, 777
 Address, 778
 ConnectAttempts, 778
 ConnectInterval, 778
 ConnectionDefaultRtt, 778
 ConnectionGroups, 778
 ConnectionPingInterval, 779
 ConnectionSendBuffers, 779
 ConnectionShutdownTime, 779
 ConnectionsPerGroup, 781
 ConnectionTimeout, 779
 DefaultMtu, 779
 Defaults, 781
 MaxConnections, 779
 Notify, 780
 OperationExpireTime, 780
 PacketSize, 780
 PacketSizeInBits, 781
 Simulation, 780
 SocketRecvBuffer, 780
 SocketSendBuffer, 780
NetConfigPointer
 Simulation, 730
NetConnectFailedReason
 Fusion.Sockets, 58
NetDisconnectReason
 Fusion.Sockets, 59
NetPacketType
 Fusion.Sockets, 59
Network
 NetworkProjectConfig, 496
NetworkArray
 NetworkArray< T >, 257
 NetworkArray< T >, 255
 Clear, 257
 CopyFrom, 258
 CopyTo, 259
 Get, 260
 GetEnumerator, 260
 Length, 261
 NetworkArray, 257
 operator NetworkArrayReadOnly< T >, 260
 Set, 260
 this[int index], 261
 ToArray, 260
 ToListString, 261
 ToReadOnly, 261
 ToString, 261
 NetworkArray< T >.Enumerator, 262
 Current, 263
 Dispose, 263
 Enumerator, 262
 MoveNext, 263
 Reset, 263
 NetworkArrayExtensions, 264
 GetRef< T >, 264
 IndexOf< T >, 264
NetworkArrayReadOnly< T >, 265
 Length, 266
 this[int index], 266
NetworkAssemblyIgnoreAttribute, 266
NetworkAssemblyWeavedAttribute, 266
NetworkBehaviour, 266
 ChangedTick, 287
 CopyBackingFieldsToState, 270
 CopyStateFrom, 270
 CopyStateToBackingFields, 270
 Despawned, 270
 DynamicWordCount, 288
 FixedUpdateNetwork, 271
 GetArrayReader< T >, 271
 GetBehaviourReader< T >, 272, 273
 GetBehaviourReader< TBehaviour, TProperty >, 273
 GetChangeDetector, 274
 GetDictionaryReader< K, V >, 274, 275
 GetInput< T >, 275
 GetLinkListReader< T >, 276
 GetLocalAuthorityMask, 277
 GetPropertyReader< T >, 277
 GetPropertyReader< TBehaviour, TProperty >, 278
 HasInputAuthority, 288
 HasStateAuthority, 288
 Id, 288
 IsProxy, 288
 MakeInitializer< K, V >, 279
 MakeInitializer< T >, 279
 MakePtr< T >, 279
 MakeRef< T >, 280, 281

NetworkDeserialize, 281
 NetworkSerialize, 282
 NetworkUnwrap, 283
 NetworkWrap, 283
 offset, 287
 operator NetworkBehaviourId, 283
 ReinterpretState< T >, 285
 ReplicateTo, 285, 286
 ReplicateToAll, 286
 ResetState, 286
 Spawned, 286
 StateBuffer, 288
 StateBufferIsValid, 289
 TryGetSnapshotsBuffers, 287
 NetworkBehaviour.ArrayReader< T >, 289
 Read, 289
 NetworkBehaviour.BehaviourReader< T >, 290
 Read, 290
 T, 291
 NetworkBehaviour.ChangeDetector, 291
 DetectChanges, 292, 293
 Init, 293
 SimulationState, 292
 SnapshotFrom, 292
 SnapshotTo, 292
 Source, 292
 NetworkBehaviour.ChangeDetector.Enumerable, 294
 Changed, 294
 GetEnumerator, 294
 NetworkBehaviour.ChangeDetector.Enumerator, 295
 Current, 296
 MoveNext, 295
 Reset, 295
 NetworkBehaviour.DictionaryReader< K, V >, 296
 Read, 296
 NetworkBehaviour.LinkListReader< T >, 297
 Read, 297
 NetworkBehaviour.PropertyReader< T >, 298
 PropertyReader, 298
 Read, 299
 T, 299
 NetworkBehaviourBuffer, 299
 Length, 306
 operator bool, 300
 Read, 301, 303
 Read< T >, 303, 305
 ReinterpretState< T >, 305
 this[int index], 306
 Tick, 306
 Valid, 307
 NetworkBehaviourBufferInterpolator, 307
 Alpha, 317
 Angle, 309
 Behaviour, 317
 Bool, 309, 310
 Float, 310
 From, 317
 Int, 312
 NetworkBehaviourBufferInterpolator, 308
 operator bool, 312
 Quaternion, 313
 Select< T >, 313, 314
 To, 317
 Valid, 317
 Vector2, 314, 315
 Vector3, 315, 316
 Vector4, 316
 NetworkBehaviourId, 318
 Behaviour, 321
 Equals, 319
 GetHashCode, 319
 IsValid, 321
 None, 321
 Object, 321
 operator!=, 320
 operator==, 320
 SIZE, 321
 ToString, 320
 NetworkBehaviourUtils, 322
 CloneArray< T >, 324
 CopyFromNetworkArray< T >, 324
 CopyFromNetworkDictionary< D, K, V >, 325
 CopyFromNetworkList< T >, 325
 GetMetaData, 326
 GetRpcStaticIndexOrThrow, 326
 GetStaticWordCount, 327
 GetWordCount, 327
 HasStaticWordCount, 327
 InitializeNetworkArray< T >, 328
 InitializeNetworkDictionary< D, K, V >, 328
 InitializeNetworkList< T >, 329
 InternalOnDestroy, 330
 InternalOnDisable, 330
 InternalOnEnable, 330
 InvokeRpc, 336
 MakeSerializableDictionary< K, V >, 331
 NotifyLocalSimulationNotAllowedToSendRpc, 331
 NotifyLocalTargetedRpcCulled, 332
 NotifyNetworkUnwrapFailed< T >, 332
 NotifyNetworkWrapFailed< T >, 332, 333
 NotifyRpcPayloadSizeExceeded, 333
 NotifyRpcTargetUnreachable, 333
 RegisterMetaData, 334
 RegisterRpcInvokeDelegates, 334
 ShouldRegisterRpcInvokeDelegates, 334
 ThrowIfBehaviourNotInitialized, 335
 TryGetRpcInvokeDelegateArray, 335
 TryGetRpcStaticInvokeDelegate, 335
 NetworkBehaviourUtils.ArrayInitializer< T >, 336
 operator NetworkArray< T >, 337
 operator NetworkLinkedList< T >, 337
 NetworkBehaviourUtils.DictionaryInitializer< K, V >, 338
 operator NetworkDictionary< K, V >, 338
 NetworkBehaviourUtils.MetaData, 339
 NetworkBehaviourWeavedAttribute, 339

NetworkBehaviourWeavedAttribute, 339
WordCount, 340
NetworkBool, 340
Equals, 341
GetHashCode, 342
NetworkBool, 341
operator bool, 342
operator NetworkBool, 342
ToString, 342
NetworkButtons, 343
Bits, 353
Equals, 344, 345
GetHashCode, 345
GetPressed, 345
GetReleased, 346
IsSet, 346
IsSet< T >, 346
NetworkButtons, 344, 352
Set, 348
Set< T >, 348
SetAllDown, 349
SetAllUp, 349
SetDown, 349
SetDown< T >, 349
SetUp, 350
SetUp< T >, 350
WasPressed, 351
WasPressed< T >, 351
WasReleased, 351
WasReleased< T >, 352
NetworkConditions
 NetworkProjectConfig, 496
NetworkConfiguration, 353
 ClientToClientWithServerProxy, 354
 ClientToServer, 354
 ConnectAttempts, 355
 ConnectInterval, 355
 ConnectionDefaultRtt, 356
 ConnectionPingInterval, 356
 ConnectionShutdownTime, 355
 ConnectionTimeout, 355
 Init, 354
 MtuDefault, 356
 ReliableDataTransferModes, 355
 ReliableDataTransfers, 354
 SocketRecvBufferSize, 356
 SocketSendBufferSize, 356
NetworkConnected
 Simulation, 725
NetworkDelegates, 357
NetworkDeserialize
 NetworkBehaviour, 281
NetworkDeserializeMethodAttribute, 357
NetworkDictionary
 NetworkDictionary< K, V >, 360
NetworkDictionary< K, V >, 358
 Add, 360
 Capacity, 364
 Clear, 361
 ContainsKey, 361
 ContainsValue, 361
 Count, 364
 Get, 361
 GetEnumerator, 361
 META_WORD_COUNT, 364
 NetworkDictionary, 360
 operator NetworkDictionaryReadOnly< K, V >, 362
 Remove, 362
 Set, 363
 this[K key], 364
 ToReadOnly, 363
 TryGet, 363
 NetworkDictionary< K, V >.Enumerator, 364
 Current, 366
 Dispose, 365
 MoveNext, 365
 Reset, 365
 NetworkDictionaryReadOnly< K, V >, 366
 Capacity, 368
 Count, 368
 Get, 367
 TryGet, 367
NetworkDisconnected
 Simulation, 725
NetworkedAttribute, 368
 Default, 369
 NetworkedAttribute, 369
NetworkedBehaviours
 NetworkObject, 415
NetworkedWeavedAttribute, 369
 NetworkedWeavedAttribute, 370
 WordCount, 370
 WordOffset, 370
NetworkEvents, 370
 NetworkEvents.ConnectFailedEvent, 372
 NetworkEvents.ConnectRequestEvent, 372
 NetworkEvents.CustomAuthenticationResponse, 372
 NetworkEvents.DisconnectFromServerEvent, 372
 NetworkEvents.HostMigrationEvent, 373
 NetworkEvents.InputEvent, 373
 NetworkEvents.InputPlayerEvent, 373
 NetworkEvents.ObjectEvent, 373
 NetworkEvents.ObjectPlayerEvent, 374
 NetworkEvents.PlayerEvent, 374
 NetworkEvents.ReliableDataEvent, 374
 NetworkEvents.ReliableProgressEvent, 374
 NetworkEvents.RunnerEvent, 375
 NetworkEvents.SessionListUpdateEvent, 375
 NetworkEvents.ShutdownEvent, 375
 NetworkEvents.SimulationMessageEvent, 375
NetworkId, 376
 ALIGNMENT, 380
 BLOCK_SIZE, 380
 Comparer, 381
 CompareTo, 377

Equals, 378
GetHashCode, 378
IsReserved, 381
IsValid, 381
operator bool, 378
operator!=, 379
operator==, 379
Raw, 381
Read, 379
SIZE, 381
ToNamePrefixString, 379
ToString, 380
Write, 380
NetworkId.EqualityComparer, 382
 Equals, 382
 GetHashCode, 382
NetworkIdIsObjectName
 NetworkProjectConfig, 496
NetworkInput, 382
 Convert< T >, 383
 Data, 385
 Get< T >, 383
 Is< T >, 384
 IsValid, 385
 Set< T >, 384
 TryGet< T >, 384
 TrySet< T >, 384
 Type, 385
 WordCount, 385
NetworkInputUtils, 386
 GetMaxWordCount, 386
 GetType, 386
 GetTypeKey, 387
 GetWordCount, 387
NetworkInputWeavedAttribute, 387
 NetworkInputWeavedAttribute, 388
 WordCount, 388
NetworkLinkedList
 NetworkLinkedList< T >, 390
NetworkLinkedList< T >, 388
 Add, 391
 Capacity, 394
 Clear, 391
 Contains, 392
 Count, 394
 ELEMENT_WORDS, 394
 Get, 392
 GetEnumerator, 392
 IndexOf, 392
 META_WORDS, 394
 NetworkLinkedList, 390
 Remap, 393
 Remove, 393
 Set, 393
 this[int index], 394
NetworkLinkedList< T >.Enumerator, 395
 Current, 396
 Dispose, 395
MoveNext, 396
Reset, 396
NetworkLinkedListReadOnly< T >, 397
 Capacity, 400
 Contains, 399
 Count, 400
 ELEMENT_WORDS, 400
 Get, 399
 IndexOf, 399
 META_WORDS, 400
 this[int index], 400
NetworkLoadSceneParameters, 401
 Equals, 402
 GetHashCode, 402
 IsActiveOnLoad, 404
 IsLocalPhysics2D, 404
 IsLocalPhysics3D, 404
 IsSingleLoad, 404
 LoadId, 403
 LoadSceneMode, 404
 LoadSceneParameters, 404
 LocalPhysicsMode, 405
 operator!=, 402
 operator==, 403
 ToString, 403
NetworkMecanimAnimator, 405
 Animator, 407
 ApplyTiming, 407
 DynamicWordCount, 407
 SetTrigger, 406
NetworkObject, 408
 AssignInputAuthority, 410
 Awake, 410
 CopyStateFrom, 410, 411
 Flags, 415
 GetLocalAuthorityMask, 411
 GetWordCount, 411
 HasInputAuthority, 416
 HasStateAuthority, 416
 Id, 416
 InputAuthority, 416
 IsInSimulation, 417
 IsProxy, 417
 IsResume, 415
 IsSpawnable, 417
 IsValid, 417
 LastReceiveTick, 417
 Name, 417
 NestedObjects, 415
 NetworkedBehaviours, 415
 NetworkTypeId, 415
 NetworkUnwrap, 412
 NetworkWrap, 412
 OnDestroy, 413
 operator NetworkId, 413
 PriorityCallback, 415
 PriorityLevelDelegate, 413
 ReleaseStateAuthority, 413

RemoveInputAuthority, 414
RenderSource, 418
RenderTime, 418
RenderTimeframe, 418
ReplicateTo, 416
ReplicateToDelegate, 414
RequestStateAuthority, 414
Runner, 418
SetPlayerAlwaysInterested, 414
SortKey, 416
StateAuthority, 418
NetworkObjectAcquireResult
 Fusion, 42
NetworkObjectFlags
 Fusion, 43
NetworkObjectFlagsExtensions, 419
 GetVersion, 419
 IsIgnored, 419
 IsVersionCurrent, 421
 SetCurrentVersion, 421
 SetIgnored, 421
NetworkObjectGuid, 422
 ALIGNMENT, 429
 CompareTo, 425
 Empty, 429
 Equals, 425
 GetHashCode, 426
 IsValid, 429
 NetworkObjectGuid, 423, 424
 operator Guid, 426
 operator NetworkObjectGuid, 426
 operator NetworkPrefabRef, 427
 operator!=, 427
 operator==, 427
 Parse, 427
 RawGuidValue, 429
 SIZE, 429
 ToString, 428
 ToUnityGuidString, 428
 TryParse, 428
NetworkObjectGuid.EqualityComparer, 430
 Equals, 430
 GetHashCode, 430
NetworkObjectHeader, 430
 _reserved, 435
 BehaviourCount, 435
 ByteCount, 437
 Equals, 432
 Flags, 435
 GetBehaviourChangedTickArray, 432
 GetDataPointer, 433
 GetDataWordCount, 433
 GetHashCode, 433
 GetMainNetworkTRSPData, 434
 HasMainNetworkTRSP, 434
 Id, 435
 InputAuthority, 436
 NestingKey, 436
 NestingRoot, 436
 operator!=, 434
 operator==, 434
 PLAYER_DATA_WORD, 436
 SIZE, 436
 StateAuthority, 436
 ToString, 435
 Type, 437
 WordCount, 437
 WORDS, 437
 NetworkObjectHeaderFlags
 Fusion, 43
 NetworkObjectHeaderPtr, 437
 Id, 438
 Ptr, 438
 Type, 438
 NetworkObjectInitializerUnity, 439
 InitializeNetworkState, 439
 NetworkObjectMeta, 439
 Id, 440
 InputAuthority, 440
 StateAuthority, 440
 Type, 440
 NetworkObjectNestingKey, 440
 ALIGNMENT, 443
 Equals, 442
 GetHashCode, 442
 IsNone, 443
 IsValid, 444
 NetworkObjectNestingKey, 441
 SIZE, 443
 ToString, 443
 Value, 443
 NetworkObjectNestingKey.EqualityComparer, 444
 Equals, 444
 GetHashCode, 445
 NetworkObjectPrefabData, 445
 Guid, 445
 NetworkObjectProviderDummy, 446
 NetworkObjectReleaseContext, 446
 IsBeingDestroyed, 447
 IsNestedObject, 447
 NetworkObjectReleaseContext, 447
 Object, 448
 ToString, 447
 Typeld, 448
 NetworkObjectSortKeyComparer, 448
 Compare, 448
 Instance, 449
 NetworkObjectSpawnDelegate
 Fusion, 53
 NetworkObjectSpawnException, 449
 Message, 450
 NetworkObjectSpawnException, 449
 Status, 450
 Typeld, 450
 NetworkObjectStatisticsManager, 796
 ClearMonitoredNetworkObjects, 796

GetNetworkObjectStatistics, 796
MonitorNetworkObjectStatistics, 797
NetworkObjectStatisticsSnapshot, 797
InBandwidth, 798
InPackets, 798
OutBandwidth, 798
OutPackets, 798
NetworkObjectTypeId, 450
_value0, 456
_value1, 456
ALIGNMENT, 456
AsCustom, 456
AsInternalStructId, 457
AsPrefabId, 457
AsSceneObjectId, 457
Comparer, 457
Equals, 452, 453
FromCustom, 453
FromPrefabId, 453
FromSceneRefAndObjectIndex, 454
FromStruct, 454
GetHashCode, 455
IsCustom, 458
IsNone, 458
IsPrefab, 458
IsSceneObject, 458
IsStruct, 458
IsValid, 458
Kind, 459
MAX_SCENE_OBJECT_INDEX, 456
operator NetworkObjectTypeId, 455
operator!=, 455
operator==, 455
PlayerData, 459
SIZE, 456
ToString, 455
NetworkObjectTypeId.EqualityComparer, 459
Equals, 459
GetHashCode, 460
NetworkPhysicsInfo, 461
SIZE, 461
TimeScale, 461
WORD_COUNT, 461
NetworkPrefabAcquireContext, 462
Data, 463
DontDestroyOnLoad, 463
HasHeader, 464
IsSynchronous, 463
Meta, 463
NetworkPrefabAcquireContext, 462
PrefabId, 463
NetworkPrefabAttribute, 464
NetworkPrefabId, 464
ALIGNMENT, 468
AsIndex, 469
CompareTo, 466
Equals, 466
FromIndex, 466
FromRaw, 467
GetHashCode, 467
IsNone, 469
IsValid, 469
MAX_INDEX, 468
operator!=, 467
operator==, 467
RawValue, 468
SIZE, 468
ToString, 467, 468
NetworkPrefabId.EqualityComparer, 469
Equals, 470
GetHashCode, 470
NetworkPrefabInfo, 470
Data, 471
HasHeader, 471
Header, 471
IsSynchronous, 471
Prefab, 471
NetworkPrefabRef, 472
ALIGNMENT, 478
CompareTo, 474
Empty, 479
Equals, 475
GetHashCode, 475
IsValid, 479
NetworkPrefabRef, 473, 474
operator Guid, 476
operator NetworkObjectGuid, 476
operator NetworkPrefabRef, 476
operator!=, 477
operator==, 477
Parse, 477
RawGuidValue, 479
SIZE, 479
ToString, 477, 478
ToUnityGuidString, 478
TryParse, 478
NetworkPrefabRef.EqualityComparer, 479
Equals, 480
GetHashCode, 480
NetworkPrefabTable, 480
AddInstance, 482
AddSource, 482
Clear, 482
Contains, 483
GetEntries, 483
GetGuid, 483
GetId, 483
GetInstanceCount, 484
GetSource, 484, 485
IsAcquired, 485
Load, 485
Options, 488
Prefabs, 488
RemoveInstance, 486
TryAddSource, 486
Unload, 487

UnloadAll, 487
UnloadUnreferenced, 487
Version, 488
NetworkPrefabTableGetPrefabResult
 Fusion, 44
NetworkPrefabTableOptions, 488
 Default, 489
 UnloadPrefabOnReleasingLastInstance, 489
 UnloadUnusedPrefabsOnShutdown, 489
NetworkProjectConfig, 489
 AssembliesToWeave, 494
 BuildTypes, 494
 CheckNetworkedPropertiesBeingEmpty, 494
 CheckRpcAttributeUsage, 494
 CurrentTypeid, 494
 CurrentVersion, 494
 DefaultResourceName, 495
 Deserialize, 492
 EncryptionConfig, 495
 EnqueueIncompleteSynchronousSpawns, 495
 GetExecutionOrder, 493
 Global, 498
 Heap, 495
 HideNetworkObjectInactivityGuard, 495
 HostMigration, 495
 InvokeRenderInBatchMode, 496
 LagCompensation, 496
 Multiple, 492
 Network, 496
 NetworkConditions, 496
 NetworkIdIsObjectName, 496
 None, 492
 NullChecksForNetworkedProperties, 496
 PeerMode, 497
 PeerModes, 491
 PrefabTable, 497
 ReplicationFeatures, 492
 Scheduling, 492
 SchedulingAndInterestManagement, 492
 Serialize, 493
 Simulation, 497
 Single, 492
 TimeSynchronizationOverride, 497
 ToString, 493
 Typeid, 497
 UnloadGlobal, 493
 UseSerializableDictionary, 497
 Version, 498
NetworkProjectConfigAsset, 498
 BehaviourMeta, 500
 Config, 500
 Global, 500
 IsGlobalLoaded, 501
 PrefabOptions, 500
 Prefabs, 500
 TryGetGlobal, 499
 UnloadGlobal, 499
NetworkProjectConfigAsset.SerializableSimulationBehaviourMeta,
 501
 ExecutionOrder, 501
 Type, 501
NetworkReceiveDone
 Simulation, 726
NetworkRNG, 502
 MAX, 506
 NetworkRNG, 503
 Next, 503
 NextExclusive, 503
 NextInt32, 504
 NextSingle, 504
 NextSingleExclusive, 504
 NextUInt32, 504
 Peek, 507
 RangeExclusive, 505
 RangeInclusive, 505, 506
 SIZE, 506
 ToString, 506
NetworkRpcStaticWeavedInvokerAttribute, 507
 Key, 508
 NetworkRpcStaticWeavedInvokerAttribute, 507
NetworkRpcWeavedInvokerAttribute, 508
 Key, 509
 NetworkRpcWeavedInvokerAttribute, 508
 Sources, 509
 Targets, 509
NetworkRunner, 509
 ActivePlayers, 562
 AddCallbacks, 519
 AddGlobal, 519
 AddPlayerAreaOfInterest, 519
 Attach, 519, 520
 AuthenticationValues, 562
 BuildType, 562
 BuildTypes, 518
 CanSpawn, 562
 ClearPlayerAreaOfInterest, 520
 Config, 563
 CurrentConnectionType, 563
 Debug, 518
 DeltaTime, 563
 Despawn, 520
 DestroySingleton< T >, 521
 Disconnect, 521
 EnsureRunnerScenesActive, 521
 Exists, 522
 FindObject, 522
 GameMode, 563
 GetAllBehaviours, 522
 GetAllBehaviours< T >, 523
 GetAreaOfInterestGizmoData, 524
 GetInputForPlayer< T >, 524
 GetInstanceEnumerator, 524
 GetInterfaceListHead, 524
 GetInterfaceListNext, 525
 GetInterfaceListPrev, 525

GetInterfaceListsCount, 525
GetPhysicsScene, 526
GetPhysicsScene2D, 526
GetPlayerActorId, 526
GetPlayerConnectionToken, 527
GetPlayerConnectionType, 527
GetPlayerObject, 527
GetPlayerRtt, 528
GetPlayerUserId, 528
GetRawInputForPlayer, 528
GetResumeSnapshotNetworkObjects, 528
GetResumeSnapshotNetworkSceneObjects, 529
GetRpcTargetStatus, 529
GetRunnerForObject, 529
GetRunnerForScene, 530
GetSingleton< T >, 530
HasSingleton< T >, 530
Instances, 563
InstantiateInRunnerScene, 530, 531
InstantiateInRunnerScene< T >, 531
InvokeSceneLoadDone, 531
InvokeSceneLoadStart, 532
IsClient, 563
IsCloudReady, 564
IsConnectedToServer, 564
IsFirstTick, 564
IsForward, 564
IsInterestedIn, 532
IsLastTick, 564
IsPlayer, 564
IsPlayerValid, 532
IsResimulation, 565
IsResume, 565
IsRunning, 565
IsSceneAuthority, 565
IsSceneManagerBusy, 565
IsServer, 565
IsSharedModeMasterClient, 566
IsShutdown, 566
IsSinglePlayer, 566
IsStarting, 566
JoinSessionLobby, 532
LagCompensation, 566
LatestServerTick, 566
LoadScene, 533, 534
LobbyInfo, 567
LocalAlpha, 567
LocalPlayer, 567
LocalRenderTime, 567
MakeDontDestroyOnLoad, 534
Mode, 567
MoveGameObjectToSameScene, 535
MoveGameObjectToScene, 535
MoveToRunnerScene, 536
MoveToRunnerScene< T >, 536
NATTType, 567
ObjectAcquired, 570
ObjectDelegate, 536
ObjectProvider, 568
OnBeforeSpawned, 536
Prefabs, 568
ProvideInput, 568
PushHostMigrationSnapshot, 537
RegisterSceneObjects, 537
Release, 518
RemoteRenderTime, 568
RemoveCallbacks, 537
RemoveGlobal, 538
RenderInternal, 538
Running, 518
SceneManager, 568
SendReliableDataToPlayer, 538
SendReliableDataToServer, 538
SendRpc, 539
SessionInfo, 568
SetAreaOfInterestCellSize, 539
SetAreaOfInterestGrid, 541
SetBehaviourReplicateTo, 541
SetBehaviourReplicateToAll, 541
SetIsSimulated, 542
SetMasterClient, 542
SetPlayerAlwaysInterested, 542
SetPlayerObject, 543
Shutdown, 518, 543
SimulationTime, 569
SimulationUnityScene, 569
SinglePlayerContinue, 543
SinglePlayerPause, 543, 544
Spawn, 544–546
Spawn< T >, 547
SpawnAsync, 547–550
SpawnAsync< T >, 551
Stage, 569
StartGame, 551
Starting, 518
State, 569
States, 518
Tick, 569
TickRate, 569
TicksExecuted, 570
Topology, 570
TryFindBehaviour, 552
TryFindBehaviour< T >, 552
TryFindObject, 553
TryGetFusionStatistics, 553
TryGetInputForPlayer< T >, 553
TryGetNetworkedBehaviourFromNetworkedObjectRef< T >, 554
TryGetNetworkedBehaviourId, 554
TryGetObjectRefFromNetworkedBehaviour, 555
TryGetPhysicsInfo, 555
TryGetPlayerObject, 555
TryGetSceneInfo, 556
TrySetPhysicsInfo, 556
TrySpawn, 557–559
TrySpawn< T >, 560

UnloadScene, 560
UpdateInternal, 562
UserId, 570
NetworkRunnerCallbackArgs, 570
NetworkRunnerCallbackArgs.ConnectRequest, 571
 Accept, 571
 Refuse, 571
 RemoteAddress, 572
 Waiting, 572
NetworkRunnerUpdaterDefault, 572
 RegisterInPlayerLoop, 573
 RenderSettings, 574
 UnregisterFromPlayerLoop, 573
 UpdateSettings, 574
NetworkRunnerUpdaterDefault.NetworkRunnerRender, 574
NetworkRunnerUpdaterDefault.NetworkRunnerUpdate, 574
NetworkRunnerUpdaterDefaultInvokeSettings, 575
 AddMode, 577
 Equals, 575, 576
 GetHashCode, 576
 operator!=, 576
 operator==, 577
 ReferencePlayerLoopSystem, 578
 ToString, 577
NetworkSceneAsyncOp, 578
 AddOnCompleted, 579
 Error, 582
 FromAsyncOperation, 579
 FromCompleted, 580
 FromCoroutine, 580
 FromError, 581
 FromTask, 581
 GetAwaiter, 581
 IsDone, 582
 IsValid, 582
 SceneRef, 582
NetworkSceneAsyncOp.Awaiter, 583
 Awaiter, 583
 GetResult, 584
 IsCompleted, 584
 OnCompleted, 584
NetworkSceneInfo, 584
 AddSceneRef, 586
 Equals, 586
 GetHashCode, 587
 IndexOf, 587
 MaxScenes, 588
 operator NetworkSceneInfo, 587
 RemoveSceneRef, 588
 SceneCount, 589
 SceneParams, 589
 Scenes, 589
 SIZE, 588
 ToString, 588
 Version, 589
 WORD_COUNT, 589
NetworkSceneInfoChangeSource
 Fusion, 44
NetworkSceneInfoDefaultFlags
 Fusion, 44
NetworkSceneLoadId, 590
 Equals, 590, 591
 GetHashCode, 591
 NetworkSceneLoadId, 590
 Value, 591
NetworkSceneObjectId, 592
 Equals, 592, 593
 GetHashCode, 593
 IsValid, 594
 ObjectId, 593
 Scene, 594
 SceneLoadId, 594
 ToString, 593
NetworkSerialize
 NetworkBehaviour, 282
NetworkSerializeMethodAttribute, 594
 MaxSize, 595
NetworkSimulationConfiguration, 595
 AdditionalJitter, 596
 AdditionalLoss, 597
 Clone, 596
 Create, 596
 DelayMax, 597
 DelayMin, 597
 DelayPeriod, 597
 DelayShape, 597
 DelayThreshold, 597
 Enabled, 598
 LossChanceMax, 598
 LossChanceMin, 598
 LossChancePeriod, 598
 LossChanceShape, 598
 LossChanceThreshold, 598
NetworkSpawnFlags
 Fusion, 44
NetworkSpawnOp, 599
 GetAwaiter, 600
 IsFailed, 600
 IsQueued, 600
 IsSpawned, 600
 Object, 600
 Runner, 600
 Status, 601
NetworkSpawnOp.Awaiter, 601
 Awaiter, 601
 GetResult, 602
 IsCompleted, 602
 OnCompleted, 602
NetworkSpawnStatus
 Fusion, 45
NetworkString
 NetworkString< TSize >, 606
NetworkString< TSize >, 603
 Assign, 606

Capacity, 628
 Compare, 606, 607
 Compare< TOtherSize >, 607, 608
 Contains, 608, 609
 Contains< TOtherSize >, 610
 EndsWith, 611
 EndsWith< TOtherSize >, 611
 Equals, 612, 613
 Equals< TOtherSize >, 613, 614
 Get, 614
 GetCapacity< TSize >, 615
 GetCharCount, 615
 GetEnumerator, 615
 GetHashCode, 615
 IndexOf, 616–618
 IndexOf< TOtherSize >, 619–621
 Length, 628
 NetworkString, 606
 operator NetworkString< TSize >, 621
 operator string, 622
 operator!=, 622, 623
 operator==, 623, 624
 Set, 624
 StartsWith, 625
 StartsWith< TOtherSize >, 625
 Substring, 626
 this[int index], 628
 ToLower, 627
 ToString, 627
 ToUpper, 627
 Value, 628
 NetworkStructUtils, 628
 GetWordCount< T >, 629
 NetworkStructWeavedAttribute, 629
 NetworkStructWeavedAttribute, 630
 WordCount, 630
 NetworkTransform, 630
 AutoUpdateAreaOfInterestOverride, 633
 DisableSharedModelInterpolation, 632
 SetAreaOfInterestOverride, 631
 SyncParent, 632
 SyncScale, 632
 Teleport, 632
 NetworkTRSP, 633
 Data, 635
 IsMainTRSP, 636
 Render, 634
 ResolveAOIOVERRIDE, 634
 SetAreaOfInterestOverride, 635
 SetParentTransform, 635
 State, 636
 Teleport, 635
 NetworkTRSPData, 636
 AreaOfInterestOverride, 637
 NonNetworkedParent, 639
 Parent, 637
 Position, 637
 POSITION_OFFSET, 637
 Rotation, 638
 Scale, 638
 SIZE, 638
 TeleportKey, 638
 WORDS, 638
 NetworkTypeId
 NetworkObject, 415
 NetworkTypeIdKind
 Fusion, 45
 NetworkUnwrap
 NetworkBehaviour, 283
 NetworkObject, 412
 NetworkWrap
 NetworkBehaviour, 283
 NetworkObject, 412
 Next
 NetworkRNG, 503
 Tick, 801
 NextExclusive
 NetworkRNG, 503
 NextInt32
 NetworkRNG, 504
 NextSingle
 NetworkRNG, 504
 NextSingleExclusive
 NetworkRNG, 504
 NextUInt32
 NetworkRNG, 504
 NoActiveConnections
 Fusion, 48
 NONE
 AuthorityMasks, 87
 None
 Fusion, 41–44, 48, 52
 Fusion.LagCompensation, 56
 NetworkBehaviourId, 321
 NetworkProjectConfig, 492
 PlayerRef, 648
 SceneRef, 706
 TickTimer, 825
 NonNetworkedParent
 NetworkTRSPData, 639
 Normal
 LagCompensatedHit, 213
 LagCompensationUtils.ContactData, 227
 Normalized
 Fusion, 52
 NormalizedPercentage
 Fusion, 52
 NormalizedRectAttribute, 639
 AspectRatio, 640
 InvertY, 640
 NormalizedRectAttribute, 639
 NoSimulation
 Simulation, 726
 NotCulled
 Fusion, 48
 NotEqual

Fusion, 41
NotFound
 Fusion, 44
 TickRate, 811
Notify
 NetConfig, 780
NotifyLocalSimulationNotAllowedToSendRpc
 NetworkBehaviourUtils, 331
NotifyLocalTargetedRpcCulled
 NetworkBehaviourUtils, 332
NotifyNetworkUnwrapFailed< T >
 NetworkBehaviourUtils, 332
NotifyNetworkWrapFailed< T >
 NetworkBehaviourUtils, 332, 333
NotifyRpcPayloadSizeExceeded
 NetworkBehaviourUtils, 333
NotifyRpcTargetUnreachable
 NetworkBehaviourUtils, 333
NotInvokableDuringResim
 Fusion, 47, 48
NotInvokableLocally
 Fusion, 47
NotSentBroadcastNoActiveConnections
 Fusion, 48
NotSentBroadcastNoConfirmedNorInterestedClients
 Fusion, 48
NotSentTargetClientNotAvailable
 Fusion, 48
NotSentTargetObjectNotConfirmed
 Fusion, 48
NotSentTargetObjectNotInPlayerInterest
 Fusion, 48
NotZero
 Fusion, 41
Null
 Ptr, 655
NullChecksForNetworkedProperties
 NetworkProjectConfig, 496

Object
 NetworkBehaviourId, 321
 NetworkObjectReleaseContext, 448
 NetworkSpawnOp, 600
 RpcHeader, 690
 SimulationBehaviour, 740
ObjectAcquired
 NetworkRunner, 570
ObjectCount
 Simulation, 731
ObjectDataConsistency
 SimulationConfig, 744
ObjectDelegate
 NetworkRunner, 536
ObjectId
 NetworkSceneObjectId, 593
ObjectInitializer
 StartGameArgs, 787
ObjectProvider
 NetworkRunner, 568
StartGameArgs, 787

Objects
 Simulation, 731
ObjectsAllocMemoryUsedInBytes
 FusionStatisticsSnapshot, 794
ObjectStatisticsManager
 FusionStatisticsManager, 791
Offset
 ColliderDrawInfo, 223
 Hitbox, 131
 HitboxRoot, 150
 SimulationMessage, 764
offset
 NetworkBehaviour, 287
Ok
 Fusion, 50
 StartGameResult, 790
 TickRate, 811
OnBeforeSpawned
 NetworkRunner, 536
OnChangedRenderAttribute, 640
 MethodName, 641
 OnChangedRenderAttribute, 641
OnCompleted
 NetworkSceneAsyncOp.Awaiter, 584
 NetworkSpawnOp.Awaiter, 602
OnConnectedToServer
 INetworkRunnerCallbacks, 178
OnConnectFailed
 INetworkRunnerCallbacks, 178
OnConnectRequest
 INetworkRunnerCallbacks, 178
OnCustomAuthenticationResponse
 INetworkRunnerCallbacks, 179
OnDestroy
 NetworkObject, 413
OnDisconnectedFromServer
 INetworkRunnerCallbacks, 179
OnDrawGizmos
 Hitbox, 130
 HitboxRoot, 148
OnGameStarted
 StartGameArgs, 787
OnHostMigration
 INetworkRunnerCallbacks, 179
OnInput
 INetworkRunnerCallbacks, 179
OnInputMissing
 INetworkRunnerCallbacks, 180
OnObjectEnterAOI
 INetworkRunnerCallbacks, 180
OnObjectExitAOI
 INetworkRunnerCallbacks, 180
OnPlayerJoined
 INetworkRunnerCallbacks, 181
OnPlayerLeft
 INetworkRunnerCallbacks, 181
OnReliableDataProgress

INetworkRunnerCallbacks, 181
operator OnReliableDataReceived
 INetworkRunnerCallbacks, 182
operator OnSceneInfoChanged
 INetworkSceneManager, 187
operator OnSceneLoadDone
 INetworkRunnerCallbacks, 182
operator OnSceneLoadStart
 INetworkRunnerCallbacks, 182
operator OnSessionListUpdated
 INetworkRunnerCallbacks, 182
operator OnShutdown
 INetworkRunnerCallbacks, 183
operator OnUserSimulationMessage
 INetworkRunnerCallbacks, 183
operator OpenInternet
 Fusion.Sockets.Stun, 59
operator OperationCanceled
 Fusion, 51
operator OperationExpireTime
 NetConfig, 780
operator OperationTimeout
 Fusion, 51
operator Angle
 Angle, 79, 80
operator bool
 NetworkBehaviourBuffer, 300
 NetworkBehaviourBufferInterpolator, 312
 NetworkBool, 342
 NetworkId, 378
 Ptr, 652
 SessionInfo, 714
 Tick, 801
operator double
 Angle, 80
operator float
 Angle, 80
 FloatCompressed, 123
operator FloatCompressed
 FloatCompressed, 124
operator Guid
 NetworkObjectGuid, 426
 NetworkPrefabRef, 476
operator int
 Tick, 802
operator LagCompensatedHit
 LagCompensatedHit, 211, 212
operator Mask256
 FieldsMask< T >, 110
operator NetworkArray< T >
 NetworkBehaviourUtils.ArrayInitializer< T >, 337
operator NetworkArrayReadOnly< T >
 NetworkArray< T >, 260
operator NetworkBehaviourId
 NetworkBehaviour, 283
operator NetworkBool
 NetworkBool, 342
operator NetworkDictionary< K, V >
 NetworkBehaviourUtils.DictionaryInitializer< K, V >, 338
operator NetworkDictionaryReadOnly< K, V >
 NetworkDictionary< K, V >, 362
operator NetworkId
 NetworkObject, 413
operator NetworkLinkedList< T >
 NetworkBehaviourUtils.ArrayInitializer< T >, 337
operator NetworkObjectGuid
 NetworkObjectGuid, 426
 NetworkPrefabRef, 476
operator NetworkObjectTypeId
 NetworkObjectTypeId, 455
operator NetworkPrefabRef
 NetworkObjectGuid, 427
 NetworkPrefabRef, 476
operator NetworkSceneInfo
 NetworkSceneInfo, 587
operator NetworkString< TSize >
 NetworkString< TSize >, 621
operator Quaternion
 QuaternionCompressed, 659
operator QuaternionCompressed
 QuaternionCompressed, 659
operator string
 NetworkString< TSize >, 622
operator Tick
 Tick, 802
operator Vector2
 Vector2Compressed, 838
 Vector3Compressed, 842
operator Vector2Compressed
 Vector2Compressed, 838
operator Vector3
 Vector3Compressed, 842
operator Vector3Compressed
 Vector3Compressed, 843
operator Vector4
 Vector4Compressed, 847
operator Vector4Compressed
 Vector4Compressed, 848
operator !=
 Angle, 81
 FloatCompressed, 124
 NetworkBehaviourId, 320
 NetworkId, 379
 NetworkLoadSceneParameters, 402
 NetworkObjectGuid, 427
 NetworkObjectHeader, 434
 NetworkObjectTypeId, 455
 NetworkPrefabId, 467
 NetworkPrefabRef, 477
 NetworkRunnerUpdaterDefaultInvokeSettings, 576
 NetworkString< TSize >, 622, 623
 PlayerRef, 644
 Ptr, 653
 QuaternionCompressed, 660
 SceneRef, 704

Tick, 802
 Vector2Compressed, 838
 Vector3Compressed, 844
 Vector4Compressed, 848

operator<
 Angle, 82
 Tick, 803

operator<=
 Angle, 82
 Tick, 803

operator>
 Angle, 83
 Tick, 803

operator>=
 Angle, 84
 Tick, 803

operator+
 Angle, 81
 Ptr, 653

operator-
 Angle, 81
 Ptr, 653

operator==
 Angle, 83
 FloatCompressed, 124
 NetworkBehaviourId, 320
 NetworkId, 379
 NetworkLoadSceneParameters, 403
 NetworkObjectGuid, 427
 NetworkObjectHeader, 434
 NetworkObjectTypeid, 455
 NetworkPrefabId, 467
 NetworkPrefabRef, 477
 NetworkRunnerUpdaterDefaultInvokeSettings, 577
 NetworkString< TSize >, 623, 624
 PlayerRef, 645
 Ptr, 654
 QuaternionCompressed, 660
 SceneRef, 704
 Tick, 803
 Vector2Compressed, 839
 Vector3Compressed, 844
 Vector4Compressed, 849

Optimize
 LagCompensationSettings, 246

Options
 NetworkPrefabTable, 488
 Query, 231
 QueryParams, 234

order
 UnityContextMenuAttribute, 828
 UnityDelayedAttribute, 828
 UnityHeaderAttribute, 830
 UnityMinAttribute, 831
 UnityMultilineAttribute, 831
 UnityNonReorderableAttribute, 832
 UnityRangeAttribute, 833
 UnitySpaceAttribute, 835

 UnityTooltipAttribute, 835

Origin
 RaycastQuery, 238
 RaycastQueryParams, 239

OutBandwidth
 FusionStatisticsSnapshot, 794
 NetworkObjectStatisticsSnapshot, 798

OutObjectUpdates
 FusionStatisticsSnapshot, 794

OutPackets
 FusionStatisticsSnapshot, 795
 NetworkObjectStatisticsSnapshot, 798

OverlapBox
 HitboxManager, 135, 136

OverlapSphere
 HitboxManager, 137–139

Packets
 Fusion, 52

PacketSize
 NetConfig, 780

PacketSizeInBits
 NetConfig, 781

PageCount
 HeapConfiguration, 128

PageShift
 HeapConfiguration, 128

PageSizes
 Fusion, 45

Parent
 NetworkTRSPData, 637

Parse
 NetworkObjectGuid, 427
 NetworkPrefabRef, 477

PayloadSizeExceeded
 Fusion, 47, 48

Peek
 NetworkRNG, 507

PeerMode
 NetworkProjectConfig, 497

PeerModes
 NetworkProjectConfig, 491

Pending
 TickAccumulator, 809

Penetration
 LagCompensationUtils.ContactData, 227

Percentage
 Fusion, 52

PerSecond
 Fusion, 52

PhotonCloudTimeout
 Fusion, 51

PhysX
 Fusion.LagCompensation, 56

Player
 Fusion, 46
 Query, 232
 QueryParams, 234
 SimulationInput, 747

PLAYER_DATA_WORD
 NetworkObjectHeader, 436

PlayerCount
 SessionInfo, 715
 SimulationConfig, 744
 StartGameArgs, 787

PlayerData
 NetworkObjectTypeid, 459

PlayerId
 PlayerRef, 648

PlayerJoined
 IPlayerJoined, 205

PlayerLeft
 IPlayerLeft, 206

PlayerMaxCount
 SimulationRuntimeConfig, 766

PlayerRef, 641
 AsIndex, 647
 Comparer, 647
 Equals, 643
 FromEncoded, 643
 FromIndex, 644
 GetHashCode, 644
 IsMasterClient, 647
 IsNone, 647
 IsRealPlayer, 648
 MASTER_CLIENT_RAW, 646
 MasterClient, 648
 None, 648
 operator!=, 644
 operator==, 645
 PlayerId, 648
 RawEncoded, 648
 Read, 645
 SIZE, 647
 ToString, 645
 Write, 646
 Write< T >, 646

Point
 LagCompensatedHit, 213
 LagCompensationUtils.ContactData, 227

PosBlendEnd
 InterpolatedErrorCorrectionSettings, 203

PosBlendStart
 InterpolatedErrorCorrectionSettings, 203

Position
 Hitbox, 132
 NetworkTRSPData, 637

POSITION_OFFSET
 NetworkTRSPData, 637

PositionRotation
 HitboxManager, 139, 140

PositionRotationQueryParams, 227
 Hitbox, 228
 PositionRotationQueryParams, 228
 QueryParams, 228

PosMinCorrection
 InterpolatedErrorCorrectionSettings, 203

PosTeleportDistance
 InterpolatedErrorCorrectionSettings, 204

Prefab
 Fusion, 45
 NetworkPrefabInfo, 471

PrefabId
 NetworkPrefabAcquireContext, 463

PrefabOptions
 NetworkProjectConfigAsset, 500

Prefabs
 NetworkPrefabTable, 488
 NetworkProjectConfigAsset, 500
 NetworkRunner, 568

PrefabTable
 NetworkProjectConfig, 497

PreProcessingDelegate
 Fusion.LagCompensation, 56
 Query, 232
 QueryParams, 234

PreserveInPluginAttribute, 649
 PreserveInPluginAttribute, 649

PriorityCallback
 NetworkObject, 415

PriorityLevel
 Fusion, 46

PriorityLevelDelegate
 NetworkObject, 413

ProjectConfig
 Simulation, 731

Properties
 SessionInfo, 716

PropertyName
 DefaultForPropertyAttribute, 92

PropertyReader
 NetworkBehaviour.PropertyReader< T >, 298

ProvideInput
 NetworkRunner, 568

Proxies
 Fusion, 49

PROXY
 AuthorityMasks, 88

Ptr, 650
 Address, 655
 Equals, 651, 652
 GetHashCode, 652
 NetworkObjectHeaderPtr, 438
 Null, 655
 operator bool, 652
 operator!=, 653
 operator+, 653
 operator-, 653
 operator==, 654
 SIZE, 655
 ToString, 654

Ptr.EqualityComparer, 655
 Equals, 656
 GetHashCode, 657

PushHostMigrationSnapshot

NetworkRunner, 537

Quaternion
 NetworkBehaviourBufferInterpolator, 313

QuaternionCompressed, 657
 Equals, 658
 GetHashCode, 659
 operator Quaternion, 659
 operator QuaternionCompressed, 659
 operator!=, 660
 operator==, 660
 W, 661
 wEncoded, 661
 X, 662
 xEncoded, 661
 Y, 662
 yEncoded, 661
 Z, 662
 zEncoded, 661

Query, 229
 Alpha, 231
 Check, 231
 LayerMask, 231
 Options, 231
 Player, 232
 PreProcessingDelegate, 232
 Query, 230
 Tick, 232
 TickTo, 232
 TriggerInteraction, 232
 UserArgs, 232

QueryParams, 233
 Alpha, 233
 BoxOverlapQueryParams, 219
 LayerMask, 233
 Options, 234
 Player, 234
 PositionRotationQueryParams, 228
 PreProcessingDelegate, 234
 RaycastQueryParams, 240
 SphereOverlapQueryParams, 244
 Tick, 234
 TickTo, 234
 TriggerInteraction, 234
 UserArgs, 235

Queued
 Fusion, 45

Radians
 Fusion, 52

RadiansPerSecond
 Fusion, 52

Radius
 ColliderDrawInfo, 223
 SphereOverlapQuery, 243
 SphereOverlapQueryParams, 244

RangeExclusive
 NetworkRNG, 505

RangeInclusive

NetworkRNG, 505, 506

Raw
 NetworkId, 381
 Tick, 804

RawEncoded
 PlayerRef, 648

RawGuidValue
 NetworkObjectGuid, 429
 NetworkPrefabRef, 479

RawValue
 NetworkPrefabId, 468
 SceneRef, 705

Raycast
 HitboxManager, 140–142

RaycastAll
 HitboxManager, 143, 144

RaycastAllQuery, 235
 RaycastAllQuery, 235, 236

RaycastQuery, 236
 Check, 237
 Direction, 238
 Length, 238
 Origin, 238
 RaycastQuery, 237

RaycastQueryParams, 238
 Direction, 239
 Length, 239
 Origin, 239
 QueryParams, 240
 RaycastQueryParams, 239
 StaticHitsCapacity, 240

Read
 IElementReaderWriter< T >, 165
 IUnitySurrogate, 190
 NetworkBehaviour.ArrayReader< T >, 289
 NetworkBehaviour.BehaviourReader< T >, 290
 NetworkBehaviour.DictionaryReader< K, V >, 296
 NetworkBehaviour.LinkedListReader< T >, 297
 NetworkBehaviour.PropertyReader< T >, 299
 NetworkBehaviourBuffer, 301, 303
 NetworkId, 379
 PlayerRef, 645
 RpcHeader, 688
 UnityArraySurrogate< T, ReaderWriter >, 193
 UnityDictionarySurrogate< TKeyType, TValueType, TKeyReaderWriter, TValueType, TValueReaderWriter >, 195
 UnityLinkedListSurrogate< T, ReaderWriter >, 197
 UnitySurrogateBase, 199
 UnityValueSurrogate< T, TReaderWriter >, 201

Read< T >
 NetworkBehaviourBuffer, 303, 305

ReadBoolean
 ReadWriteUtilsForWeaver, 672

ReadFloat
 ReadWriteUtils, 663

ReadInt
 SimulationMessage, 758

ReadNetworkBehaviourRef
 ReadWriteUtils, 664

ReadNetworkedObjectRef
 SimulationMessage, 758

ReadQuaternion
 ReadWriteUtils, 664

ReadRef
 IElementReaderWriter< T >, 165

ReadSize
 RpcHeader, 688

ReadStringUtf32NoHash
 ReadWriteUtilsForWeaver, 672

ReadStringUtf32WithHash
 ReadWriteUtilsForWeaver, 672

ReadStringUtf8NoHash
 ReadWriteUtilsForWeaver, 674

ReadVector2
 ReadWriteUtils, 664

ReadVector3
 ReadWriteUtils, 665
 SimulationMessage, 758

ReadVector4
 ReadWriteUtils, 665

ReadWriteUtils, 662
 ACCURACY, 669
 ReadFloat, 663
 ReadNetworkBehaviourRef, 664
 ReadQuaternion, 664
 ReadVector2, 664
 ReadVector3, 665
 ReadVector4, 665
 WriteEmptyNetworkBehaviourRef, 666
 WriteFloat, 666
 WriteNetworkBehaviourRef, 666
 WriteNullBehaviourRef, 666
 WriteQuaternion, 667
 WriteVector2, 667
 WriteVector3, 667
 WriteVector4, 669

ReadWriteUtilsForWeaver, 669
 GetByteArrayHashCode, 670
 GetByteCountUtf8NoHash, 670
 GetStringHashCode, 671
 GetWordCountString, 671
 ReadBoolean, 672
 ReadStringUtf32NoHash, 672
 ReadStringUtf32WithHash, 672
 ReadStringUtf8NoHash, 674
 VerifyRawNetworkUnwrap< T >, 674
 VerifyRawNetworkWrap< T >, 675
 WriteBoolean, 675
 WriteStringUtf32NoHash, 676
 WriteStringUtf32WithHash, 676
 WriteStringUtf8NoHash, 676

Redundancy
 SimulationConfig, 743

RedundancyUncompressed
 SimulationConfig, 743

RedundantInputs
 TimeSyncConfiguration, 826

RedundantSnapshots
 TimeSyncConfiguration, 826

ReferenceCountAdd
 SimulationMessage, 759

ReferenceCountSub
 SimulationMessage, 759

ReferencePlayerLoopSystem
 NetworkRunnerUpdaterDefaultInvokeSettings, 578

References
 SimulationMessage, 764

ReflectionUtils, 677
 GetAllNetworkBehaviourTypes, 677
 GetAllSimulationBehaviourTypes, 678
 GetAllWeavedAssemblies, 678
 GetAllWeavedNetworkBehaviourTypes, 678
 GetAllWeavedSimulationBehaviourTypes, 678
 GetAllWeaverGeneratedTypes, 679
 GetCustomAttributeOrThrow< T >, 679
 GetWeavedAttributeOrThrow, 680

Refuse
 NetworkRunnerCallbackArgs.ConnectRequest, 571

Region
 LobbyInfo, 247
 SessionInfo, 716

RegisterInPlayerLoop
 NetworkRunnerUpdaterDefault, 573

RegisterMetaData
 NetworkBehaviourUtils, 334

RegisterRpcInvokeDelegates
 NetworkBehaviourUtils, 334

RegisterSceneObjects
 NetworkRunner, 537

ReinitializeHitboxesBeforeRegistration
 HitboxRoot, 147

ReinterpretState< T >
 NetworkBehaviour, 285
 NetworkBehaviourBuffer, 305

Relayed
 Fusion, 41

Release
 INetworkAssetSource< T >, 172
 NetworkRunner, 518

ReleaseInstance
 INetworkObjectProvider, 176

ReleaseStateAuthority
 NetworkObject, 413

Reliable
 Fusion, 47

ReliableDataTransferModes
 NetworkConfiguration, 355

ReliableDataTransfers
 NetworkConfiguration, 354

Remainder
 TickAccumulator, 809

RemainingTicks

TickTimer, 823
RemainingTime
 TickTimer, 824
Remap
 NetworkLinkedList< T >, 393
Remote
 Fusion, 44, 46, 49
RemoteAddress
 NetworkRunnerCallbackArgs.ConnectRequest,
 572
RemoteAlpha
 Simulation, 731
RemotePrefabCreated
 IRemotePrefabCreated, 206
RemoteRenderTime
 NetworkRunner, 568
RemoteTick
 Simulation, 731
RemoteTickPrevious
 Simulation, 731
Remove
 NetBitBufferList, 773
 NetworkDictionary< K, V >, 362
 NetworkLinkedList< T >, 393
 SerializableDictionary< TKey, TValue >, 710
 SimulationInput.Buffer, 751
RemoveCallbacks
 NetworkRunner, 537
RemoveGlobal
 NetworkRunner, 538
RemoveHead
 NetBitBufferList, 773
RemoveInputAuthority
 NetworkObject, 414
RemoveInstance
 NetworkPrefabTable, 486
RemoveSceneRef
 NetworkSceneInfo, 588
Render
 NetworkTRSP, 634
 SimulationBehaviour, 739
RenderAttribute, 680
 Method, 681
 RenderAttribute, 681
 Source, 681
 Timeframe, 681
RenderInternal
 NetworkRunner, 538
RenderSettings
 NetworkRunnerUpdaterDefault, 574
RenderSource
 Fusion, 46
 NetworkObject, 418
RenderTime
 NetworkObject, 418
RenderTimeframe
 Fusion, 46
 NetworkObject, 418
 RenderTimeline, 682
 GetRenderBuffers, 682
RenderWeavedAttribute, 683
 RenderWeavedAttribute, 683
REPLICATE_WORD_ALIGN
 Allocator, 70
REPLICATE_WORD_SHIFT
 Allocator, 71
REPLICATE_WORD_SIZE
 Allocator, 71
ReplicateTo
 NetworkBehaviour, 285, 286
 NetworkObject, 416
ReplicateToAll
 NetworkBehaviour, 286
ReplicateToDelegate
 NetworkObject, 414
ReplicationFeatures
 NetworkProjectConfig, 492
 SimulationConfig, 744
RequestStateAuthority
 NetworkObject, 414
Reset
 FixedArray< T >.Enumerator, 119
 NetworkArray< T >.Enumerator, 263
 NetworkBehaviour.ChangeDetector.Enumerator,
 295
 NetworkDictionary< K, V >.Enumerator, 365
 NetworkLinkedList< T >.Enumerator, 396
 SerializableDictionary< TKey, TValue >, 711
ResetState
 NetworkBehaviour, 286
Resimulate
 Fusion, 51
Resimulations
 FusionStatisticsSnapshot, 795
Resolve
 TickRate, 814
ResolveAOIOVERRIDE
 NetworkTRSP, 634
ResolveNetworkPrefabSourceAttribute, 683
Result
 RpcSendResult, 697
Retry
 Fusion, 43
Root
 Hitbox, 131
RootGameObjects
 SceneLoadDoneArgs, 699
Rotation
 BoxOverlapQuery, 218
 BoxOverlapQueryParams, 220
 NetworkTRSPData, 638
RotBlendEnd
 InterpolatedErrorCorrectionSettings, 204
RotBlendStart
 InterpolatedErrorCorrectionSettings, 204
RotTeleportRadians

InterpolatedErrorCorrectionSettings, 204
RoundTripTime
 FusionStatisticsSnapshot, 795
RpcAttribute, 683
 Channel, 685
 HostMode, 685
 InvokeLocal, 686
 MaxPayloadSize, 685
RpcAttribute, 685
 Sources, 686
 Targets, 686
 TickAligned, 686
RpcChannel
 Fusion, 47
RpcHeader, 686
 Behaviour, 689
 Create, 687, 688
 Method, 690
 Object, 690
 Read, 688
 ReadSize, 688
 SIZE, 690
 ToString, 689
 Write, 689
RpcHostMode
 Fusion, 47
RpcInfo, 690
 Channel, 692
 FromLocal, 691
 FromMessage, 691
 IsInvokeLocal, 692
 Source, 692
 Tick, 692
 ToString, 692
RpcInvokeData, 693
 Delegate, 694
 Key, 694
 Sources, 694
 Targets, 694
 ToString, 693
RpcInvokeDelegate
 Fusion, 53
RpcInvokeInfo, 694
 LocalInvokeResult, 695
 SendCullResult, 695
 SendResult, 695
 ToString, 695
RpcLocallInvokeResult
 Fusion, 47
RpcSendCullResult
 Fusion, 48
RpcSendMessageResult
 Fusion, 48
RpcSendResult, 696
 MessageSize, 696
 Result, 697
 ToString, 696
RpcSources

Fusion, 49
RpcStaticInvokeDelegate
 Fusion, 53
RpcTargetAttribute, 697
 RpcTargetAttribute, 697
RpcTargets
 Fusion, 49
RpcTargetStatus
 Fusion, 49
Run
 TaskManager, 86
Runner
 NetworkObject, 418
 NetworkSpawnOp, 600
 SimulationBehaviour, 740
Running
 NetworkRunner, 518
 TickAccumulator, 809
SampleWindowSeconds
 TimeSyncConfiguration, 826
Scale
 NetworkTRSPData, 638
Scene
 NetworkSceneObjectId, 594
 SceneLoadDoneArgs, 699
 StartGameArgs, 787
SceneCount
 NetworkSceneInfo, 589
SceneCountMask
 Fusion, 44
SceneLoadDone
 ISceneLoadDone, 207
SceneLoadDoneArgs, 698
 RootGameObjects, 699
 Scene, 699
 SceneLoadDoneArgs, 698
 SceneObjects, 699
 SceneRef, 699
SceneLoadId
 NetworkSceneObjectId, 594
SceneLoadStart
 ISceneLoadStart, 208
SceneManager
 NetworkRunner, 568
 StartGameArgs, 787
SceneObject
 Fusion, 45
SceneObjects
 SceneLoadDoneArgs, 699
SceneParams
 NetworkSceneInfo, 589
SceneRef, 699
 AsIndex, 706
 AsPathHash, 706
 Equals, 701
 FLAG_ADDRESSABLE, 705
 FromIndex, 702
 FromPath, 702

FromRaw, 703
GetHashCode, 703
IsIndex, 706
IsPath, 703
IsValid, 706
NetworkSceneAsyncOp, 582
None, 706
operator!=, 704
operator==, 704
RawValue, 705
SceneLoadDoneArgs, 699
SIZE, 705
ToString, 704, 705
Scenes
 NetworkSceneInfo, 589
Scheduling
 NetworkProjectConfig, 492
SchedulingAndInterestManagement
 NetworkProjectConfig, 492
SchedulingEnabled
 SimulationConfig, 745
SchedulingWithoutAOI
 SimulationConfig, 745
ScriptHeaderBackColor
 Fusion, 50
ScriptHeaderIcon
 Fusion, 50
Seconds
 Fusion, 52
Select< T >
 NetworkBehaviourBufferInterpolator, 313, 314
Self
 Fusion, 49
SendCullResult
 RpcInvokeInfo, 695
SendDelta
 Simulation, 732
SendRate
 Simulation, 732
SendReliableDataToPlayer
 NetworkRunner, 538
SendReliableDataToServer
 NetworkRunner, 538
SendResult
 RpcInvokeInfo, 695
SendRpc
 NetworkRunner, 539
Sent
 SimulationInput, 747
SentBroadcast
 Fusion, 48
SentToServerForForwarding
 Fusion, 48
SentToTargetClient
 Fusion, 48
SerializableDictionary< TKey, TValue >, 707
 Add, 709
 Clear, 709
ContainsKey, 709
Count, 712
Create< TKey, TValue >, 710
EntryKeyPropertyPath, 712
GetEnumerator, 710
IsReadOnly, 712
ItemsPropertyPath, 712
Keys, 712
Remove, 710
Reset, 711
Store, 711
this[TKey key], 712
TryGetValue, 711
Values, 713
Wrap, 711
Serialize
 NetworkProjectConfig, 493
Server
 Fusion, 41, 51
 TickRate.Resolved, 818
ServerAlreadyInRoom
 Fusion.Protocol, 57
ServerFull
 Fusion.Sockets, 59
ServerIndex
 TickRate.Selection, 820
ServerIndexOutOfRange
 TickRate, 811
ServerInRoom
 Fusion, 50
ServerLogic
 Fusion.Protocol, 57
ServerMode
 SimulationRuntimeConfig, 766
ServerRefused
 Fusion.Sockets, 59
ServerSend
 TickRate.Resolved, 818
ServerSendDelta
 TickRate.Resolved, 819
ServerSendIndex
 TickRate.Selection, 821
ServerSendIndexOutOfRange
 TickRate, 811
ServerSendRateLargerThanTickRate
 TickRate, 811
ServerTickDelta
 TickRate.Resolved, 819
ServerTickStride
 TickRate.Resolved, 819
Service
 TaskManager, 86
SessionInfo, 713
 IsOpen, 715
 IsValid, 715
 IsVisible, 715
 MaxPlayers, 715
 Name, 715

NetworkRunner, 568
operator bool, 714
PlayerCount, 715
Properties, 716
Region, 716
ToString, 714
UpdateCustomProperties, 714
SessionLobby
 Fusion, 50
SessionName
 StartGameArgs, 788
SessionNameGenerator
 StartGameArgs, 788
SessionProperties
 StartGameArgs, 788
Set
 NetworkArray< T >, 260
 NetworkButtons, 348
 NetworkDictionary< K, V >, 363
 NetworkLinkedList< T >, 393
 NetworkString< TSize >, 624
Set< T >
 NetworkButtons, 348
 NetworkInput, 384
SetAllDown
 NetworkButtons, 349
SetAllUp
 NetworkButtons, 349
SetAreaOfInterestCellSize
 NetworkRunner, 539
SetAreaOfInterestGrid
 NetworkRunner, 541
SetAreaOfInterestOverride
 NetworkTransform, 631
 NetworkTRSP, 635
SetBehaviourReplicateTo
 NetworkRunner, 541
SetBehaviourReplicateToAll
 NetworkRunner, 541
SetCurrentVersion
 NetworkObjectFlagsExtensions, 421
SetDown
 NetworkButtons, 349
SetDown< T >
 NetworkButtons, 349
SetDummy
 SimulationMessage, 759
SetForcedAlive< T >
 DynamicHeap, 98
SetHitboxActive
 HitboxRoot, 149
SetIgnored
 NetworkObjectFlagsExtensions, 421
SetIsSimulated
 NetworkRunner, 542
SetLayer
 Hitbox, 130
SetMasterClient

NetworkRunner, 542
SetMinBoundingRadius
 HitboxRoot, 149
SetNotTickAligned
 SimulationMessage, 759
SetParentTransform
 NetworkTRSP, 635
SetPlayerAlwaysInterested
 NetworkObject, 414
 NetworkRunner, 542
SetPlayerObject
 NetworkRunner, 543
SetStatic
 SimulationMessage, 759
SetTarget
 SimulationMessage, 759
SetTrigger
 NetworkMecanimAnimator, 406
SetUnreliable
 SimulationMessage, 760
SetUp
 NetworkButtons, 350
Setup
 IDataEncryption, 108
 TaskManager, 86
setUp< T >
 NetworkButtons, 350
Shared
 Fusion, 41, 50, 52
SharedModeStateAuthLocalPlayer
 Fusion, 45
SharedModeStateAuthMasterClient
 Fusion, 45
ShouldRegisterRpcInvokeDelegates
 NetworkBehaviourUtils, 334
Shutdown
 INetworkRunnerUpdater, 184
 INetworkSceneManager, 187
 NetworkRunner, 518, 543
ShutdownReason
 Fusion, 50
 StartGameResult, 790
Simulation, 716
 ActivePlayers, 727
 AfterSimulation, 720
 AfterUpdate, 720
 BeforeFirstTick, 720
 BeforeSimulation, 720
 BeforeUpdate, 720
 Config, 727
 DeltaTime, 727
 GetAreaOfInterestGizmoData, 720
 GetInputAuthority, 721
 GetInputForPlayer, 721
 GetObjectsAndPlayersInAreaOfInterestCell, 721
 GetStateAuthority, 721
 HasAnyActiveConnections, 722
 InputCount, 727

IsClient, 727
IsFirstTick, 728
IsForward, 728
IsInputAuthority, 722
IsInterestedIn, 722
IsLastTick, 728
IsLocalPlayerFirstExecution, 728
IsLocalSimulationInputAuthority, 723
IsLocalSimulationStateAuthority, 723, 724
IsLocalSimulationStateOrInputSource, 724
IsMasterClient, 728
IsPlayer, 729
IsResimulation, 729
IsRunning, 729
IsServer, 729
IsShutdown, 729
IsSinglePlayer, 729
IsStateAuthority, 724, 725
LatestServerTick, 730
LocalAddress, 730
LocalAlpha, 730
LocalPlayer, 730
Mode, 730
NetConfig, 780
NetConfigPointer, 730
NetworkConnected, 725
NetworkDisconnected, 725
NetworkProjectConfig, 497
NetworkReceiveDone, 726
NoSimulation, 726
ObjectCount, 731
Objects, 731
ProjectConfig, 731
RemoteAlpha, 731
RemoteTick, 731
RemoteTickPrevious, 731
SendDelta, 732
SendRate, 732
Stage, 732
Tick, 732
TickDeltaDouble, 732
TickDeltaFloat, 732
TickPrevious, 733
TickRate, 733
TickStride, 733
Time, 733
Topology, 733
TryGetHostPlayer, 726
Update, 726
Simulation.AreaOfInterest, 734
CELL_SIZE, 737
GetCellSize, 734
SphereToCells, 734
ToCell, 736
ToCellCenter, 736
x, 737
SimulationBehaviour, 738
CanReceiveRenderCallback, 739
CanReceiveSimulationCallback, 739
FixedUpdateNetwork, 739
Object, 740
Render, 739
Runner, 740
SimulationBehaviourAttribute, 740
Modes, 741
Stages, 741
Topologies, 741
SimulationBehaviourListScope, 741
Dispose, 742
SimulationConfig, 742
AreaOfInterestEnabled, 745
DataConsistency, 743
Eventual, 743
Full, 743
HostMigration, 743
InputDataWordCount, 743
InputTotalWordCount, 745
InputTransferMode, 744
InputTransferModes, 743
LatestState, 743
ObjectDataConsistency, 744
PlayerCount, 744
Redundancy, 743
RedundancyUncompressed, 743
ReplicationFeatures, 744
SchedulingEnabled, 745
SchedulingWithoutAOI, 745
TickRateSelection, 744
Topology, 744
SimulationEnter
ISimulationEnter, 208
SimulationExit
ISimulationExit, 209
SimulationInput, 745
Clear, 746
CopyFrom, 746
Data, 747
Header, 747
Player, 747
Sent, 747
SimulationInput.Buffer, 747
Add, 749
Buffer, 748
Clear, 749
Contains, 749
CopySortedTo, 749
Count, 751
Full, 751
Get, 750
GetInsertTime, 750
GetLastUsedInputHeader, 751
Remove, 751
SimulationInputHeader, 752
InterpAlpha, 752
InterpFrom, 752
InterpTo, 752

SIZE, [753](#)
 Tick, [753](#)
 WORD_COUNT, [753](#)
SimulationMessage, [753](#)
 Allocate, [756](#)
 CanAllocateUserPayload, [756](#)
 Capacity, [761](#)
 Clone, [756](#)
 FLAG_DUMMY, [761](#)
 FLAG_INTERNAL, [761](#)
 FLAG_NOT_TICK_ALIGNED, [762](#)
 FLAG_REMOTE, [762](#)
 FLAG_STATIC, [762](#)
 FLAG_TARGET_PLAYER, [762](#)
 FLAG_TARGET_SERVER, [762](#)
 FLAG_UNRELIABLE, [762](#)
 FLAG_USER_FLAGS_START, [763](#)
 FLAG_USER_MESSAGE, [763](#)
 Flags, [763](#)
 FLAGS_RESERVED, [763](#)
 FLAGS_RESERVED_BITS, [763](#)
 GetData, [757](#)
 GetFlag, [757](#)
 IsTargeted, [757](#)
 IsUnreliable, [765](#)
 MAX_PAYLOAD_SIZE, [763](#)
 Offset, [764](#)
 ReadInt, [758](#)
 ReadNetworkedObjectRef, [758](#)
 ReadVector3, [758](#)
 ReferenceCountAdd, [759](#)
 ReferenceCountSub, [759](#)
 References, [764](#)
 SetDummy, [759](#)
 SetNotTickAligned, [759](#)
 SetStatic, [759](#)
 SetTarget, [759](#)
 SetUnreliable, [760](#)
 SIZE, [764](#)
 Source, [764](#)
 Target, [764](#)
 Tick, [764](#)
 ToString, [760](#)
 WriteInt, [760](#)
 WriteNetworkedObjectRef, [760](#)
 WriteVector3, [761](#)
SimulationMessagePtr, [765](#)
 Message, [765](#)
SimulationModes
 Fusion, [51](#)
SimulationRuntimeConfig, [765](#)
 HostPlayer, [766](#)
 MasterClient, [766](#)
 PlayerMaxCount, [766](#)
 ServerMode, [766](#)
 TickRate, [767](#)
 Topology, [767](#)
SimulationSpeed

FusionStatisticsSnapshot, [795](#)
SimulationStages
 Fusion, [51](#)
SimulationState
 NetworkBehaviour.ChangeDetector, [292](#)
SimulationTime
 NetworkRunner, [569](#)
SimulationTimeOffset
 FusionStatisticsSnapshot, [795](#)
SimulationUnityScene
 NetworkRunner, [569](#)
Single
 Fusion, [41](#)
 NetworkProjectConfig, [492](#)
SinglePlayerContinue
 NetworkRunner, [543](#)
SinglePlayerPause
 NetworkRunner, [543, 544](#)
SIZE

- _128, [61](#)
- _16, [62](#)
- _2, [63](#)
- _256, [64](#)
- _32, [65](#)
- _4, [66](#)
- _512, [67](#)
- _64, [68](#)
- _8, [68](#)

 Allocator.Config, [74](#)
 Angle, [84](#)
 NetworkBehaviourId, [321](#)
 NetworkId, [381](#)
 NetworkObjectGuid, [429](#)
 NetworkObjectHeader, [436](#)
 NetworkObjectNestingKey, [443](#)
 NetworkObjectTypeId, [456](#)
 NetworkPhysicsInfo, [461](#)
 NetworkPrefabId, [468](#)
 NetworkPrefabRef, [479](#)
 NetworkRNG, [506](#)
 NetworkSceneInfo, [588](#)
 NetworkTRSPData, [638](#)
 PlayerRef, [647](#)
 Ptr, [655](#)
 RpcHeader, [690](#)
 SceneRef, [705](#)
 SimulationInputHeader, [753](#)
 SimulationMessage, [764](#)
 Tick, [804](#)
 TickRate.Resolved, [818](#)
SnapshotFrom
 NetworkBehaviour.ChangeDetector, [292](#)
SnapshotHistoryDraw, [240](#)
 GetEnumerator, [240](#)
 LagCompensationDraw, [226](#)
SnapshotTo
 NetworkBehaviour.ChangeDetector, [292](#)
SocketRecvBuffer

NetConfig, 780
 SocketRecvBufferSize
 NetworkConfiguration, 356
 SocketSendBuffer
 NetConfig, 780
 SocketSendBufferSize
 NetworkConfiguration, 356
SortDistance
 LagCompensatedExt, 224
SortKey
 NetworkObject, 416
SortReference
 LagCompensatedExt, 225
Source
 NetworkBehaviour.ChangeDetector, 292
 RenderAttribute, 681
 RpcInfo, 692
 SimulationMessage, 764
SourcesHostPlayer
 Fusion, 47
SourcesServer
 Fusion, 47
Sources
 NetworkRpcWeavedInvokerAttribute, 509
 RpcAttribute, 686
 RpcInvokeData, 694
Spawn
 NetworkRunner, 544–546
Spawn< T >
 NetworkRunner, 547
SpawnAsync
 NetworkRunner, 547–550
SpawnAsync< T >
 NetworkRunner, 551
Spawned
 Fusion, 45
 ISpawned, 210
 NetworkBehaviour, 286
SpawnedByClient
 Fusion, 43
Sphere
 Fusion, 42
SphereOverlapQuery, 241
 Center, 242
 Check, 242
 Radius, 243
 SphereOverlapQuery, 241, 242
SphereOverlapQueryParams, 243
 Center, 244
 queryParams, 244
 Radius, 244
 SphereOverlapQueryParams, 243
 StaticHitsCapacity, 244
SphereRadius
 Hitbox, 131
SphereToCells
 Simulation.AreaOfInterest, 734
SquareMagnitude

 Fusion, 52
StackTrace
 StartGameResult, 790
Stage
 NetworkRunner, 569
 Simulation, 732
Stages
 SimulationBehaviourAttribute, 741
Start
 TickAccumulator, 808
StartGame
 NetworkRunner, 551
StartGameArgs, 782
 Address, 784
 AuthValues, 784
 Config, 784
 ConnectionToken, 784
 CustomCallbackInterfaces, 784
 CustomLobbyName, 785
 CustomPhotonAppSettings, 785
 CustomPublicAddress, 785
 CustomSTUNServer, 785
 DisableNATPunchthrough, 785
 EnableClientSessionCreation, 785
 GameMode, 786
 HostMigrationResume, 786
 HostMigrationToken, 786
 IsOpen, 786
 isVisible, 786
 MatchmakingMode, 786
 ObjectInitializer, 787
 ObjectProvider, 787
 OnGameStarted, 787
 PlayerCount, 787
 Scene, 787
 SceneManager, 787
 SessionName, 788
 SessionNameGenerator, 788
 SessionProperties, 788
 StartGameCancellationToken, 788
 ToString, 783
 Updater, 788
 UseCachedRegions, 788
 UseDefaultPhotonCloudPorts, 789
StartGameCancellationToken
 StartGameArgs, 788
StartGameResult, 789
 ErrorMessage, 790
 Ok, 790
 ShutdownReason, 790
 StackTrace, 790
 ToString, 789
Starting
 NetworkRunner, 518
StartNew
 TickAccumulator, 809
StartsWith
 NetworkString< TSize >, 625

StartsWith< TOtherSize >
 NetworkString< TSize >, 625

STATE
 AuthorityMasks, 88

State
 NetworkRunner, 569
 NetworkTRSP, 636

StateAuthority
 Fusion, 49
 NetworkObject, 418
 NetworkObjectHeader, 436
 NetworkObjectMeta, 440

StateAuthorityChanged
 IStateAuthorityChanged, 210

StateBuffer
 NetworkBehaviour, 288

StateBufferIsValid
 NetworkBehaviour, 289

StateReceiveDelta
 FusionStatisticsSnapshot, 795

States
 NetworkRunner, 518

StaticHitsCapacity
 BoxOverlapQueryParams, 220
 RaycastQueryParams, 240
 SphereOverlapQueryParams, 244

Status
 NetworkObjectSpawnException, 450
 NetworkSpawnOp, 601

Stop
 TickAccumulator, 809

Store
 SerializableDictionary< TKey, TValue >, 711

Struct
 Fusion, 43

StructArray
 Fusion, 43

StunServers.StunServer, 781

Substring
 NetworkString< TSize >, 626

SubtickAccuracy
 Fusion, 42

Success
 Fusion, 43, 44

SurrogateType
 FixedBufferPropertyAttribute, 120

Symmetric
 Fusion.Sockets.Stun, 59

SyncParent
 NetworkTransform, 632

SyncScale
 NetworkTransform, 632

T
 NetworkBehaviour.BehaviourReader< T >, 291
 NetworkBehaviour.PropertyReader< T >, 299

TagetPlayerIsNotLocal
 Fusion, 47

Target
 SimulationMessage, 764

TargetPlayerIsLocalButRpcIsNotInvokableLocally
 Fusion, 48

TargetPlayerIsNotLocal
 Fusion, 47

TargetPlayerUnreachable
 Fusion, 48

Targets
 NetworkRpcWeavedInvokerAttribute, 509
 RpcAttribute, 686
 RpcInvokeData, 694

TargetTick
 TickTimer, 825

TaskManager, 85
 ContinueWhenAll, 85
 Run, 86
 Service, 86
 Setup, 86

Teleport
 INetworkTRSPTeleport, 189
 NetworkTransform, 632
 NetworkTRSP, 635

TeleportKey
 NetworkTRSPData, 638

this[int index]
 FixedSize< T >, 117
 INetworkArray, 170
 NetworkArray< T >, 261
 NetworkArrayReadOnly< T >, 266
 NetworkBehaviourBuffer, 306
 NetworkLinkedList< T >, 394
 NetworkLinkedListReadOnly< T >, 400
 NetworkString< TSize >, 628
 TickRate, 816

this[K key]
 NetworkDictionary< K, V >, 364

this[TKey key]
 SerializableDictionary< TKey, TValue >, 712

ThrowIfBehaviourNotInitialized
 NetworkBehaviourUtils, 335

Tick, 798
 ALIGNMENT, 804
 CompareTo, 800
 Equals, 800
 GetHashCode, 801
 NetworkBehaviourBuffer, 306
 NetworkRunner, 569
 Next, 801
 operator bool, 801
 operator int, 802
 operator Tick, 802
 operator!=, 802
 operator<, 803
 operator<=, 803
 operator>, 803
 operator>=, 803
 operator==, 803
 Query, 232

QueryParams, 234
Raw, 804
RpcInfo, 692
Simulation, 732
SimulationInputHeader, 753
SimulationMessage, 764
SIZE, 804
ToString, 803
Tick.EqualityComparer, 804
Equals, 805
GetHashCode, 805
Tick.RelationalComparer, 805
Compare, 806
TickAccumulator, 806
AddTicks, 807
AddTime, 807
Alpha, 808
ConsumeTick, 808
Pending, 809
Remainder, 809
Running, 809
Start, 808
StartNew, 809
Stop, 809
TimeScale, 809
TickAligned
 RpcAttribute, 686
TickDeltaDouble
 Simulation, 732
TickDeltaFloat
 Simulation, 732
TickPrevious
 Simulation, 733
TickRate, 810
 Available, 816
 ClampSelection, 812
 Client, 816
 ClientSendIndexOutOfRange, 811
 Count, 816
 Error, 811
 Get, 812
 GetDivisor, 812
 GetTickRate, 813
 Init, 813
 InvalidTickRate, 811
 IsValid, 813, 814
 NetworkRunner, 569
 NotFound, 811
 Ok, 811
 Resolve, 814
 ServerIndexOutOfRange, 811
 ServerSendIndexOutOfRange, 811
 ServerSendRateLargerThanTickRate, 811
 Simulation, 733
 SimulationRuntimeConfig, 767
 this[int index], 816
 ToArray, 815
 Validate, 815
 ValidateResult, 811
 ValidateSelection, 815
 TickRate.Resolved, 817
 Client, 818
 ClientSend, 818
 ClientSendDelta, 819
 ClientTickDelta, 819
 ClientTickStride, 819
 Server, 818
 ServerSend, 818
 ServerSendDelta, 819
 ServerTickDelta, 819
 ServerTickStride, 819
 SIZE, 818
 WORDS, 818
 TickRate.Selection, 820
 Client, 820
 ClientSendIndex, 820
 ServerIndex, 820
 ServerSendIndex, 821
 TickRateSelection
 SimulationConfig, 744
 Ticks
 Fusion, 52
 TicksExecuted
 NetworkRunner, 570
 TicksPerSecond
 Fusion, 52
 TickStride
 Simulation, 733
 TickTimer, 821
 CreateFromSeconds, 822
 CreateFromTicks, 822
 Expired, 823
 ExpiredOrNotRunning, 823
 IsRunning, 824
 None, 825
 RemainingTicks, 823
 RemainingTime, 824
 TargetTick, 825
 ToString, 824
 TickTo
 Query, 232
 QueryParams, 234
 Time
 Simulation, 733
 Timeframe
 RenderAttribute, 681
 Timeout
 Fusion.Sockets, 59
 TimeResets
 FusionStatisticsSnapshot, 796
 TimeScale
 NetworkPhysicsInfo, 461
 TickAccumulator, 809
 TimeSyncConfiguration, 825
 MaxLateInputs, 825
 MaxLateSnapshots, 826

RedundantInputs, 826
 RedundantSnapshots, 826
 SampleWindowSeconds, 826
 TimeSynchronizationOverride
 NetworkProjectConfig, 497
 To
 Fusion, 46
 NetworkBehaviourBufferInterpolator, 317
 ToArray
 FixedArray< T >, 116
 NetworkArray< T >, 260
 TickRate, 815
 ToCell
 Simulation.AreaOfInterest, 736
 ToCellCenter
 Simulation.AreaOfInterest, 736
 ToStringString
 FixedArray< T >, 116
 NetworkArray< T >, 261
 ToLower
 NetworkString< TSize >, 627
 ToNamePrefixString
 NetworkId, 379
 Topologies
 Fusion, 52
 SimulationBehaviourAttribute, 741
 Topology
 NetworkRunner, 570
 Simulation, 733
 SimulationConfig, 744
 SimulationRuntimeConfig, 767
 ToReadOnly
 NetworkArray< T >, 261
 NetworkDictionary< K, V >, 363
 ToString
 Allocator.Config, 73
 Angle, 84
 FixedArray< T >, 117
 HeapConfiguration, 127
 NetworkArray< T >, 261
 NetworkBehaviourId, 320
 NetworkBool, 342
 NetworkId, 380
 NetworkLoadSceneParameters, 403
 NetworkObjectGuid, 428
 NetworkObjectHeader, 435
 NetworkObjectNestingKey, 443
 NetworkObjectReleaseContext, 447
 NetworkObjectTypeid, 455
 NetworkPrefabId, 467, 468
 NetworkPrefabRef, 477, 478
 NetworkProjectConfig, 493
 NetworkRNG, 506
 NetworkRunnerUpdaterDefaultInvokeSettings, 577
 NetworkSceneInfo, 588
 NetworkSceneObjectId, 593
 NetworkString< TSize >, 627
 PlayerRef, 645
 Ptr, 654
 RpcHeader, 689
 Rpclnfo, 692
 RpclnvokeData, 693
 RpclnvokeInfo, 695
 RpclSendResult, 696
 SceneRef, 704, 705
 SessionInfo, 714
 SimulationMessage, 760
 StartGameArgs, 783
 StartGameResult, 789
 Tick, 803
 TickTimer, 824
 TotalHitboxes
 HitboxManager, 145
 ToUnityGuidString
 NetworkObjectGuid, 428
 NetworkPrefabRef, 478
 ToUpper
 NetworkString< TSize >, 627
 TriggerInteraction
 Query, 232
 QueryParams, 234
 TryAddSource
 NetworkPrefabTable, 486
 TryFindBehaviour
 NetworkRunner, 552
 TryFindBehaviour< T >
 NetworkRunner, 552
 TryFindObject
 NetworkRunner, 553
 TryGet
 NetworkDictionary< K, V >, 363
 NetworkDictionaryReadOnly< K, V >, 367
 TryGet< T >
 NetworkInput, 384
 TryGetBehaviour< T >
 Behaviour, 89
 TryGetFusionStatistics
 NetworkRunner, 553
 TryGetGlobal
 NetworkProjectConfigAsset, 499
 TryGetHostPlayer
 Simulation, 726
 TryGetInputForPlayer< T >
 NetworkRunner, 553
 TryGetNetworkedBehaviourFromNetworkedObjectRef<
 T >
 NetworkRunner, 554
 TryGetNetworkedBehaviourId
 NetworkRunner, 554
 TryGetObjectRefFromNetworkedBehaviour
 NetworkRunner, 555
 TryGetPhysicsInfo
 NetworkRunner, 555
 TryGetPhysicsScene2D
 INetworkSceneManager, 187
 TryGetPhysicsScene3D

INetworkSceneManager, 188
TryGetPlayerObject
 NetworkRunner, 555
TryGetRpcInvokeDelegateArray
 NetworkBehaviourUtils, 335
TryGetRpcStaticInvokeDelegate
 NetworkBehaviourUtils, 335
TryGetSceneInfo
 NetworkRunner, 556
TryGetSnapshotsBuffers
 NetworkBehaviour, 287
TryGetValue
 SerializableDictionary< TKey, TValue >, 711
TryParse
 NetworkObjectGuid, 428
 NetworkPrefabRef, 478
TrySet< T >
 NetworkInput, 384
TrySetPhysicsInfo
 NetworkRunner, 556
TrySpawn
 NetworkRunner, 557–559
TrySpawn< T >
 NetworkRunner, 560
Type
 ColliderDrawInfo, 223
 FixedBufferPropertyAttribute, 120
 Hitbox, 131
 LagCompensatedHit, 214
 NetworkInput, 385
 NetworkObjectHeader, 437
 NetworkObjectHeaderPtr, 438
 NetworkObjectMeta, 440
 NetworkProjectConfigAsset.SerializableSimulationBehaviourMeta, 501
 UnitySpaceAttribute, 835
 UnitySpaceAttribute, 834
 UnitySurrogateBase, 198
 Init, 199
 Read, 199
 Write, 199
 UnityTooltipAttribute, 835
 order, 835
 UnityTooltipAttribute, 835
 UnityValueSurrogate< T, TReaderWriter >, 200
 DataProperty, 202
 Init, 201
 Read, 201
 Write, 201
 Unload
 NetworkPrefabTable, 487
 UnloadAll
 NetworkPrefabTable, 487
 UnloadGlobal
 NetworkProjectConfig, 493
 NetworkProjectConfigAsset, 499
 UnloadPrefabOnReleasingLastInstance
 NetworkPrefabTableOptions, 489
 UnloadScene
 INetworkSceneManager, 188
 DataProperty, 196
 Init, 195
 Read, 195
 Write, 195
 UnityEngine, 60
 UnityEngine.SceneManagement, 60
 UnityEngine.Scripting, 60
 UnityEngine.Serialization, 60
 UnityFormerlySerializedAsAttribute, 828
 UnityFormerlySerializedAsAttribute, 829
 UnityHeaderAttribute, 829
 order, 830
 UnityHeaderAttribute, 829
 UnityLinkedListSurrogate< T, ReaderWriter >, 196
 DataProperty, 198
 Init, 197
 Read, 197
 Write, 197
 UnityMinAttribute, 830
 order, 831
 UnityMinAttribute, 830
 UnityMultilineAttribute, 831
 order, 831
 UnityNonReorderableAttribute, 832
 order, 832
 UnityNonSerializedAttribute, 832
 UnityPlayerLoopSystemAddMode
 Fusion, 52
 UnityRangeAttribute, 832
 order, 833
 UnityRangeAttribute, 833
 UnitySerializeField, 833
 UnitySerializeReference, 834
 UnitySpaceAttribute, 834
 UnitySpaceAttribute, 834
 UnitySurrogateBase, 198
 Init, 199
 Read, 199
 Write, 199
 UnityTooltipAttribute, 835
 order, 835
 UnityTooltipAttribute, 835
 UnityValueSurrogate< T, TReaderWriter >, 200
 DataProperty, 202
 Init, 201
 Read, 201
 Write, 201
 Unload
 NetworkPrefabTable, 487
 UnloadAll
 NetworkPrefabTable, 487
 UnloadGlobal
 NetworkProjectConfig, 493
 NetworkProjectConfigAsset, 499
 UnloadPrefabOnReleasingLastInstance
 NetworkPrefabTableOptions, 489
 UnloadScene
 INetworkSceneManager, 188

NetworkRunner, 560
 UnloadUnreferenced
 NetworkPrefabTable, 487
 UnloadUnusedPrefabsOnShutdown
 NetworkPrefabTableOptions, 489
 Unreachable
 Fusion, 49
 UnregisterFromPlayerLoop
 NetworkRunnerUpdaterDefault, 573
 Unreliable
 Fusion, 47
 Update
 Simulation, 726
 UpdateCustomProperties
 SessionInfo, 714
 UpdateDelay
 HostMigrationConfig, 152
 UpdateInternal
 NetworkRunner, 562
 Updater
 StartGameArgs, 788
 UpdateSettings
 NetworkRunnerUpdaterDefault, 574
 UseCachedRegions
 StartGameArgs, 788
 UseDefaultPhotonCloudPorts
 StartGameArgs, 789
 UserArgs
 Query, 232
 QueryParams, 235
 UserId
 NetworkRunner, 570
 UseSerializableDictionary
 NetworkProjectConfig, 497

V1
 Fusion, 43

Valid
 NetworkBehaviourBuffer, 307
 NetworkBehaviourBufferInterpolator, 317

Validate
 TickRate, 815

ValidateResult
 TickRate, 811

ValidateSelection
 TickRate, 815

Value
 NetworkObjectNestingKey, 443
 NetworkSceneLoadId, 591
 NetworkString< TSize >, 628

valueEncoded
 FloatCompressed, 125

Values
 SerializableDictionary< TKey, TValue >, 713

Vector2
 NetworkBehaviourBufferInterpolator, 314, 315

Vector2Compressed, 836
 Equals, 837
 GetHashCode, 837

operator Vector2, 838
 operator Vector2Compressed, 838
 operator!=, 838
 operator==, 839
 X, 840
 xEncoded, 839
 Y, 840
 yEncoded, 839

Vector3
 NetworkBehaviourBufferInterpolator, 315, 316

Vector3Compressed, 840
 Equals, 841
 GetHashCode, 842
 operator Vector2, 842
 operator Vector3, 842
 operator Vector3Compressed, 843
 operator!=, 844
 operator==, 844
 X, 845
 xEncoded, 844
 Y, 845
 yEncoded, 844
 Z, 845
 zEncoded, 845

Vector4
 NetworkBehaviourBufferInterpolator, 316

Vector4Compressed, 845
 Equals, 847
 GetHashCode, 847
 operator Vector4, 847
 operator Vector4Compressed, 848
 operator!=, 848
 operator==, 849
 W, 850
 wEncoded, 849
 X, 850
 xEncoded, 849
 Y, 850
 yEncoded, 849
 Z, 850
 zEncoded, 850

VerifyHash
 IDataEncryption, 108

VerifyRawNetworkUnwrap< T >
 ReadWriteUtilsForWeaver, 674

VerifyRawNetworkWrap< T >
 ReadWriteUtilsForWeaver, 675

Version
 NetworkPrefabTable, 488
 NetworkProjectConfig, 498
 NetworkScenelInfo, 589

Versioning, 649
 GetCurrentVersion, 650
 GetProductVersion, 650

W
 QuaternionCompressed, 661
 Vector4Compressed, 850

WaitForResult

INetworkAssetSource< T >, 172
Waiting
 NetworkRunnerCallbackArgs.ConnectRequest, 572
WarnIfAttribute, 851
 Message, 851
WasPressed
 NetworkButtons, 351
WasPressed< T >
 NetworkButtons, 351
WasReleased
 NetworkButtons, 351
WasReleased< T >
 NetworkButtons, 352
WeaverGeneratedAttribute, 851
wEncoded
 QuaternionCompressed, 661
 Vector4Compressed, 849
WORD_COUNT
 NetworkPhysicsInfo, 461
 NetworkSceneInfo, 589
 SimulationInputHeader, 753
WordCount
 DefaultForPropertyAttribute, 92
 NetworkBehaviourWeavedAttribute, 340
 NetworkedWeavedAttribute, 370
 NetworkInput, 385
 NetworkInputWeavedAttribute, 388
 NetworkObjectHeader, 437
 NetworkStructWeavedAttribute, 630
WordOffset
 DefaultForPropertyAttribute, 92
 NetworkedWeavedAttribute, 370
WORDS
 NetworkObjectHeader, 437
 NetworkTRSPData, 638
 TickRate.Resolved, 818
Wrap
 SerializableDictionary< TKey, TValue >, 711
Write
 IElementReaderWriter< T >, 165
 IUnitySurrogate, 191
 NetworkId, 380
 PlayerRef, 646
 RpcHeader, 689
 UnityArraySurrogate< T, ReaderWriter >, 193
 UnityDictionarySurrogate< TKeyType, TKeyReaderWriter, TValueType, TValueReaderWriter >, 195
 UnityLinkedListSurrogate< T, ReaderWriter >, 197
 UnitySurrogateBase, 199
 UnityValueSurrogate< T, TReaderWriter >, 201
Write< T >
 PlayerRef, 646
WriteBoolean
 ReadWriteUtilsForWeaver, 675
WriteEmptyNetworkBehaviourRef
 ReadWriteUtils, 666
 WriteFloat
 ReadWriteUtils, 666
WriteInt
 SimulationMessage, 760
WriteNetworkBehaviourRef
 ReadWriteUtils, 666
WriteNetworkedObjectRef
 SimulationMessage, 760
WriteNullBehaviourRef
 ReadWriteUtils, 666
WriteQuaternion
 ReadWriteUtils, 667
WriteStringUtf32NoHash
 ReadWriteUtilsForWeaver, 676
WriteStringUtf32WithHash
 ReadWriteUtilsForWeaver, 676
WriteStringUtf8NoHash
 ReadWriteUtilsForWeaver, 676
WriteVector2
 ReadWriteUtils, 667
WriteVector3
 ReadWriteUtils, 667
 SimulationMessage, 761
WriteVector4
 ReadWriteUtils, 669
X
 QuaternionCompressed, 662
 Vector2Compressed, 840
 Vector3Compressed, 845
 Vector4Compressed, 850
x
 Simulation.AreaOfInterest, 737
xEncoded
 QuaternionCompressed, 661
 Vector2Compressed, 839
 Vector3Compressed, 844
 Vector4Compressed, 849
Y
 QuaternionCompressed, 662
 Vector2Compressed, 840
 Vector3Compressed, 845
 Vector4Compressed, 850
yEncoded
 QuaternionCompressed, 661
 Vector2Compressed, 839
 Vector3Compressed, 844
 Vector4Compressed, 849
Z
 QuaternionCompressed, 662
 Vector3Compressed, 845
 Vector4Compressed, 850
zEncoded
 QuaternionCompressed, 661
 Vector3Compressed, 845
 Vector4Compressed, 850