

Droid Soccer

1.00

Generated by Doxygen 1.8.11

Contents

1	Module Index	1
1.1	Modules	1
2	Namespace Index	3
2.1	Packages	3
3	Hierarchical Index	5
3.1	Class Hierarchy	5
4	Class Index	9
4.1	Class List	9
5	File Index	13
5.1	File List	13
6	Module Documentation	17
6.1	Public API	17
6.1.1	Detailed Description	19
6.1.2	Enumeration Type Documentation	19
6.1.2.1	DisconnectCause	19
6.1.2.2	PeerState	20
6.1.2.3	PhotonLogLevel	20
6.1.2.4	PhotonNetworkingMessage	21
6.1.2.5	PhotonTargets	24
6.1.3	Function Documentation	25
6.1.3.1	OnPhotonSerializeView(PhtonStream stream, PhotonMessageInfo info)	25
6.2	Optional Gui Elements	26
6.2.1	Detailed Description	26

7 Namespace Documentation	27
7.1 ExitGames Namespace Reference	27
7.2 ExitGames.Client Namespace Reference	27
7.3 ExitGames.Client.DemoParticle Namespace Reference	27
7.4 ExitGames.Client.GUI Namespace Reference	27
7.4.1 Enumeration Type Documentation	28
7.4.1.1 GizmoType	28
7.5 ExitGames.Client.Photon Namespace Reference	28
7.5.1 Enumeration Type Documentation	29
7.5.1.1 EventCaching	29
7.5.1.2 JoinMode	29
7.5.1.3 MatchmakingMode	30
7.5.1.4 PropertyTypeFlag	30
7.5.1.5 ReceiverGroup	30
7.6 Photon Namespace Reference	30
7.6.1 Typedef Documentation	31
7.6.1.1 Hashtable	31
7.7 Rotorz Namespace Reference	31
7.8 Rotorz.ReorderableList Namespace Reference	31
7.9 Rotorz.ReorderableList.Internal Namespace Reference	31
7.10 UnityEngine Namespace Reference	31
7.11 UnityEngine.SceneManagement Namespace Reference	31

8 Class Documentation	33
8.1 AccountService Class Reference	33
8.1.1 Member Enumeration Documentation	33
8.1.1.1 Origin	33
8.1.2 Constructor & Destructor Documentation	34
8.1.2.1 AccountService()	34
8.1.3 Member Function Documentation	34
8.1.3.1 RegisterByEmail(string email, Origin origin)	34
8.1.3.2 RegisterByEmailAsync(string email, Origin origin, Action< AccountService > callback=null)	34
8.1.3.3 Validator(object sender, X509Certificate certificate, X509Chain chain, SslPolicyErrors policyErrors)	34
8.1.4 Property Documentation	34
8.1.4.1 Appld	34
8.1.4.2 Message	34
8.1.4.3 ReturnCode	34
8.2 ExitGames.Client.Photon.ActorProperties Class Reference	35
8.2.1 Detailed Description	35
8.2.2 Member Data Documentation	35
8.2.2.1 IsInactive	35
8.2.2.2 PlayerName	35
8.2.2.3 UserId	35
8.3 AuthenticationValues Class Reference	35
8.3.1 Detailed Description	36
8.3.2 Constructor & Destructor Documentation	36
8.3.2.1 AuthenticationValues()	36
8.3.2.2 AuthenticationValues(string userId)	36
8.3.3 Member Function Documentation	37
8.3.3.1 AddAuthParameter(string key, string value)	37
8.3.3.2 SetAuthPostData(string stringData)	37
8.3.3.3 SetAuthPostData(byte[] byteData)	37

8.3.3.4	ToString()	37
8.3.4	Property Documentation	37
8.3.4.1	AuthGetParameters	37
8.3.4.2	AuthPostData	38
8.3.4.3	AuthType	38
8.3.4.4	Token	38
8.3.4.5	UserId	38
8.4	ConnectAndJoinRandom Class Reference	38
8.4.1	Detailed Description	39
8.4.2	Member Function Documentation	39
8.4.2.1	OnConnectedToMaster()	39
8.4.2.2	OnFailedToConnectToPhoton(DisconnectCause cause)	39
8.4.2.3	OnJoinedLobby()	39
8.4.2.4	OnJoinedRoom()	39
8.4.2.5	OnPhotonRandomJoinFailed()	39
8.4.2.6	Start()	39
8.4.2.7	Update()	39
8.4.3	Member Data Documentation	39
8.4.3.1	AutoConnect	39
8.4.3.2	Version	39
8.5	ExitGames.Client.Photon.LoadbalancingPeer.EnterRoomParams Class Reference	39
8.5.1	Member Data Documentation	40
8.5.1.1	CreateIfNotExists	40
8.5.1.2	Lobby	40
8.5.1.3	OnGameServer	40
8.5.1.4	PlayerProperties	40
8.5.1.5	RoomName	40
8.5.1.6	RoomOptions	40
8.6	ExitGames.Client.Photon.ErrorCode Class Reference	40
8.6.1	Detailed Description	41

8.6.2	Member Data Documentation	41
8.6.2.1	AlreadyMatched	41
8.6.2.2	AuthenticationTicketExpired	41
8.6.2.3	CustomAuthenticationFailed	41
8.6.2.4	GameClosed	41
8.6.2.5	GameDoesNotExist	42
8.6.2.6	GameFull	42
8.6.2.7	GameIdAlreadyExists	42
8.6.2.8	InternalServerError	42
8.6.2.9	InvalidAuthentication	42
8.6.2.10	InvalidOperation	42
8.6.2.11	InvalidOperationCode	42
8.6.2.12	InvalidRegion	42
8.6.2.13	MaxCcuReached	43
8.6.2.14	NoRandomMatchFound	43
8.6.2.15	Ok	43
8.6.2.16	OperationNotAllowedInCurrentState	43
8.6.2.17	PluginMismatch	43
8.6.2.18	PluginReportedError	43
8.6.2.19	ServerFull	43
8.6.2.20	UserBlocked	44
8.7	ExitGames.Client.Photon.EventCode Class Reference	44
8.7.1	Detailed Description	45
8.7.2	Member Data Documentation	45
8.7.2.1	AppStats	45
8.7.2.2	AzureNodeInfo	45
8.7.2.3	CacheSliceChanged	45
8.7.2.4	ErrorInfo	45
8.7.2.5	GameList	45
8.7.2.6	GameListUpdate	45

8.7.2.7	Join	45
8.7.2.8	Leave	46
8.7.2.9	LobbyStats	46
8.7.2.10	Match	46
8.7.2.11	PropertiesChanged	46
8.7.2.12	QueueState	46
8.7.2.13	SetProperties	46
8.8	Extensions Class Reference	46
8.8.1	Detailed Description	47
8.8.2	Member Function Documentation	47
8.8.2.1	AlmostEquals(this Vector3 target, Vector3 second, float sqrMagnitudePrecision)	47
8.8.2.2	AlmostEquals(this Vector2 target, Vector2 second, float sqrMagnitudePrecision)	47
8.8.2.3	AlmostEquals(this Quaternion target, Quaternion second, float maxAngle)	47
8.8.2.4	AlmostEquals(this float target, float second, float floatDiff)	48
8.8.2.5	Contains(this int[] target, int nr)	48
8.8.2.6	GetPhotonView(this UnityEngine.GameObject go)	48
8.8.2.7	GetPhotonViewsInChildren(this UnityEngine.GameObject go)	48
8.8.2.8	Merge(this IDictionary target, IDictionary addHash)	48
8.8.2.9	MergeStringKeys(this IDictionary target, IDictionary addHash)	48
8.8.2.10	StripKeysWithNullValues(this IDictionary original)	49
8.8.2.11	StripToStringKeys(this IDictionary original)	49
8.8.2.12	ToStringFull(this IDictionary origin)	49
8.9	FriendInfo Class Reference	49
8.9.1	Detailed Description	50
8.9.2	Member Function Documentation	50
8.9.2.1	ToString()	50
8.9.3	Property Documentation	50
8.9.3.1	IsInRoom	50
8.9.3.2	IsOnline	50
8.9.3.3	Name	50

8.9.3.4	Room	50
8.10	GameObjectExtensions Class Reference	50
8.10.1	Detailed Description	51
8.10.2	Member Function Documentation	51
8.10.2.1	GetActive(this GameObject target)	51
8.11	ExitGames.Client.Photon.GamePropertyKey Class Reference	51
8.11.1	Detailed Description	52
8.11.2	Member Data Documentation	52
8.11.2.1	CleanupCacheOnLeave	52
8.11.2.2	IsOpen	52
8.11.2.3	IsVisible	52
8.11.2.4	MasterClientId	52
8.11.2.5	MaxPlayers	52
8.11.2.6	PlayerCount	52
8.11.2.7	PropsListedInLobby	52
8.11.2.8	Removed	53
8.12	ExitGames.Client.GUI.GizmoTypeDrawer Class Reference	53
8.12.1	Member Function Documentation	53
8.12.1.1	Draw(Vector3 center, GizmoType type, Color color, float size)	53
8.13	HelpURL Class Reference	53
8.13.1	Detailed Description	53
8.13.2	Constructor & Destructor Documentation	54
8.13.2.1	HelpURL(string url)	54
8.14	HighlightOwnedGameObj Class Reference	54
8.14.1	Member Data Documentation	54
8.14.1.1	Offset	54
8.14.1.2	PointerPrefab	54
8.15	InputToEvent Class Reference	54
8.15.1	Detailed Description	55
8.15.2	Member Data Documentation	55

8.15.2.1	DetectPointedAtGameObject	55
8.15.2.2	Dragging	55
8.15.2.3	inputHitPos	55
8.15.3	Property Documentation	55
8.15.3.1	DragVector	55
8.15.3.2	goPointedAt	55
8.16	InRoomChat Class Reference	55
8.16.1	Member Function Documentation	56
8.16.1.1	AddLine(string newLine)	56
8.16.1.2	Chat(string newLine, PhotonMessageInfo mi)	56
8.16.1.3	OnGUI()	56
8.16.1.4	Start()	56
8.16.2	Member Data Documentation	56
8.16.2.1	AlignBottom	56
8.16.2.2	ChatRPC	56
8.16.2.3	GuiRect	56
8.16.2.4	IsVisible	56
8.16.2.5	messages	56
8.17	InRoomRoundTimer Class Reference	56
8.17.1	Detailed Description	57
8.17.2	Member Function Documentation	57
8.17.2.1	OnGUI()	57
8.17.2.2	OnJoinedRoom()	57
8.17.2.3	OnMasterClientSwitched(PhotonPlayer newMasterClient)	57
8.17.2.4	OnPhotonCustomRoomPropertiesChanged(Hashtable propertiesThatChanged)	57
8.17.3	Member Data Documentation	58
8.17.3.1	secondsbeforeend	58
8.17.3.2	SecondsPerTurn	58
8.17.3.3	starting_severtime	58
8.17.3.4	StartTime	58

8.17.3.5	TextPos	58
8.17.3.6	timetostart	58
8.18	IPunCallbacks Interface Reference	58
8.18.1	Detailed Description	59
8.18.2	Member Function Documentation	60
8.18.2.1	OnConnectedToMaster()	60
8.18.2.2	OnConnectedToPhoton()	60
8.18.2.3	OnConnectionFail(DisconnectCause cause)	60
8.18.2.4	OnCreatedRoom()	60
8.18.2.5	OnCustomAuthenticationFailed(string debugMessage)	61
8.18.2.6	OnDisconnectedFromPhoton()	62
8.18.2.7	OnFailedToConnectToPhoton(DisconnectCause cause)	62
8.18.2.8	OnJoinedLobby()	62
8.18.2.9	OnJoinedRoom()	62
8.18.2.10	OnLeftLobby()	63
8.18.2.11	OnLeftRoom()	63
8.18.2.12	OnLobbyStatisticsUpdate()	63
8.18.2.13	OnMasterClientSwitched(PhotonPlayer newMasterClient)	63
8.18.2.14	OnOwnershipRequest(object[] viewAndPlayer)	63
8.18.2.15	OnPhotonCreateRoomFailed(object[] codeAndMsg)	64
8.18.2.16	OnPhotonCustomRoomPropertiesChanged(Hashtable propertiesThatChanged)	64
8.18.2.17	OnPhotonInstantiate(PhotonMessageInfo info)	64
8.18.2.18	OnPhotonJoinRoomFailed(object[] codeAndMsg)	64
8.18.2.19	OnPhotonMaxCccuReached()	65
8.18.2.20	OnPhotonPlayerConnected(PhotonPlayer newPlayer)	65
8.18.2.21	OnPhotonPlayerDisconnected(PhotonPlayer otherPlayer)	65
8.18.2.22	OnPhotonPlayerPropertiesChanged(object[] playerAndUpdatedProps)	65
8.18.2.23	OnPhotonRandomJoinFailed(object[] codeAndMsg)	66
8.18.2.24	OnReceivedRoomListUpdate()	66
8.18.2.25	OnUpdatedFriendList()	66

8.18.2.26 OnWebRpcResponse(OperationResponse response)	67
8.19 IPunObservable Interface Reference	67
8.19.1 Detailed Description	67
8.20 IPunPrefabPool Interface Reference	68
8.20.1 Detailed Description	68
8.20.2 Member Function Documentation	68
8.20.2.1 Destroy(GameObject gameObject)	68
8.20.2.2 Instantiate(string prefabId, Vector3 position, Quaternion rotation)	68
8.21 ManualPhotonViewAllocator Class Reference	69
8.21.1 Member Function Documentation	69
8.21.1.1 AllocateManualPhotonView()	69
8.21.1.2 InstantiateRpc(int viewID)	69
8.21.2 Member Data Documentation	69
8.21.2.1 Prefab	69
8.22 Photon.MonoBehaviour Class Reference	70
8.22.1 Detailed Description	70
8.22.2 Property Documentation	70
8.22.2.1 networkView	70
8.22.2.2 photonView	70
8.23 MoveByKeys Class Reference	70
8.23.1 Detailed Description	71
8.23.2 Member Function Documentation	71
8.23.2.1 FixedUpdate()	71
8.23.2.2 IsGrounded()	71
8.23.2.3 Start()	71
8.23.3 Member Data Documentation	71
8.23.3.1 dir	71
8.23.3.2 dirFix	71
8.23.3.3 disableTime	71
8.23.3.4 drag	72

8.23.3.5	JumpForce	72
8.23.3.6	JumpTimeout	72
8.23.3.7	lastPowerup	72
8.23.3.8	maxPowerups	72
8.23.3.9	powerText	72
8.23.3.10	powerups	72
8.23.3.11	rad	72
8.23.3.12	thrust	72
8.24	OnAwakeUsePhotonView Class Reference	72
8.24.1	Member Function Documentation	72
8.24.1.1	OnAwakeRPC()	72
8.24.1.2	OnAwakeRPC(byte myParameter)	72
8.25	OnClickDestroy Class Reference	73
8.25.1	Detailed Description	73
8.25.2	Member Function Documentation	73
8.25.2.1	DestroyRpc()	73
8.25.2.2	OnClick()	73
8.25.3	Member Data Documentation	73
8.25.3.1	DestroyByRpc	73
8.26	OnClickInstantiate Class Reference	74
8.26.1	Member Data Documentation	74
8.26.1.1	InstantiateType	74
8.26.1.2	Prefab	74
8.26.1.3	showGui	74
8.27	OnClickLoadSomething Class Reference	74
8.27.1	Detailed Description	75
8.27.2	Member Enumeration Documentation	75
8.27.2.1	ResourceTypeOption	75
8.27.3	Member Function Documentation	75
8.27.3.1	OnClick()	75

8.27.4	Member Data Documentation	75
8.27.4.1	ResourceToLoad	75
8.27.4.2	ResourceTypeToLoad	75
8.28	OnJoinedInstantiate Class Reference	75
8.28.1	Member Function Documentation	76
8.28.1.1	OnJoinedRoom()	76
8.28.2	Member Data Documentation	76
8.28.2.1	PlayerCamera	76
8.28.2.2	PositionOffset	76
8.28.2.3	PrefabsToInstantiate	76
8.28.2.4	SpawnPosition	76
8.29	OnStartDelete Class Reference	76
8.29.1	Detailed Description	76
8.30	ExitGames.Client.Photon.OperationCode Class Reference	76
8.30.1	Detailed Description	77
8.30.2	Member Data Documentation	77
8.30.2.1	Authenticate	77
8.30.2.2	ChangeGroups	78
8.30.2.3	CreateGame	78
8.30.2.4	ExchangeKeysForEncryption	78
8.30.2.5	FindFriends	78
8.30.2.6	GetLobbyStats	78
8.30.2.7	GetProperties	78
8.30.2.8	GetRegions	78
8.30.2.9	Join	78
8.30.2.10	JoinGame	78
8.30.2.11	JoinLobby	78
8.30.2.12	JoinRandomGame	78
8.30.2.13	Leave	79
8.30.2.14	LeaveLobby	79

8.30.2.15 RaiseEvent	79
8.30.2.16 SetProperties	79
8.30.2.17 WebRpc	79
8.31 ExitGames.Client.Photon.LoadbalancingPeer.OpJoinRandomRoomParams Class Reference	79
8.31.1 Member Data Documentation	80
8.31.1.1 ExpectedCustomRoomProperties	80
8.31.1.2 ExpectedMaxPlayers	80
8.31.1.3 MatchingType	80
8.31.1.4 SqlLobbyFilter	80
8.31.1.5 TypedLobby	80
8.32 ExitGames.Client.Photon.ParameterCode Class Reference	80
8.32.1 Detailed Description	83
8.32.2 Member Data Documentation	83
8.32.2.1 ActorList	83
8.32.2.2 ActorNr	83
8.32.2.3 Add	83
8.32.2.4 Address	83
8.32.2.5 ApplicationId	83
8.32.2.6 AppVersion	83
8.32.2.7 AzureLocalNodeId	84
8.32.2.8 AzureMasterNodeId	84
8.32.2.9 AzureNodeInfo	84
8.32.2.10 Broadcast	84
8.32.2.11 Cache	84
8.32.2.12 CacheSliceIndex	84
8.32.2.13 CheckUserOnJoin	84
8.32.2.14 CleanupCacheOnLeave	84
8.32.2.15 ClientAuthenticationData	84
8.32.2.16 ClientAuthenticationParams	84
8.32.2.17 ClientAuthenticationType	85

8.32.2.18 Code	85
8.32.2.19 CustomEventContent	85
8.32.2.20 Data	85
8.32.2.21 EmptyRoomTTL	85
8.32.2.22 EventForward	85
8.32.2.23 ExpectedValues	85
8.32.2.24 FindFriendsRequestList	85
8.32.2.25 FindFriendsResponseOnlineList	85
8.32.2.26 FindFriendsResponseRoomIdList	86
8.32.2.27 GameCount	86
8.32.2.28 GameList	86
8.32.2.29 GameProperties	86
8.32.2.30 Group	86
8.32.2.31 Info	86
8.32.2.32 IsComingBack	86
8.32.2.33 IsInactive	86
8.32.2.34 JoinMode	86
8.32.2.35 LobbyName	87
8.32.2.36 LobbyStats	87
8.32.2.37 LobbyType	87
8.32.2.38 MasterClientId	87
8.32.2.39 MasterPeerCount	87
8.32.2.40 MatchMakingType	87
8.32.2.41 PeerCount	87
8.32.2.42 PlayerProperties	87
8.32.2.43 PlayerTTL	87
8.32.2.44 PluginName	88
8.32.2.45 Plugins	88
8.32.2.46 PluginVersion	88
8.32.2.47 Position	88

8.32.2.48 Properties	88
8.32.2.49 PublishUserId	88
8.32.2.50 ReceiverGroup	88
8.32.2.51 Region	88
8.32.2.52 Remove	88
8.32.2.53 RoomName	89
8.32.2.54 Secret	89
8.32.2.55 SuppressRoomEvents	89
8.32.2.56 TargetActorNr	89
8.32.2.57 UriPath	89
8.32.2.58 UserId	89
8.32.2.59 WebRpcParameters	89
8.32.2.60 WebRpcReturnCode	89
8.32.2.61 WebRpcReturnMessage	89
8.33 PhotonAnimatorView Class Reference	90
8.33.1 Detailed Description	90
8.33.2 Member Enumeration Documentation	91
8.33.2.1 ParameterType	91
8.33.2.2 SynchronizeType	91
8.33.3 Member Function Documentation	91
8.33.3.1 DoesLayerSynchronizeTypeExist(int layerIndex)	91
8.33.3.2 DoesParameterSynchronizeTypeExist(string name)	91
8.33.3.3 GetLayerSynchronizeType(int layerIndex)	92
8.33.3.4 GetParameterSynchronizeType(string name)	92
8.33.3.5 GetSynchronizedLayers()	92
8.33.3.6 GetSynchronizedParameters()	92
8.33.3.7 SetLayerSynchronized(int layerIndex, SynchronizeType synchronizeType)	92
8.33.3.8 SetParameterSynchronized(string name, ParameterType type, SynchronizeType synchronizeType)	93
8.34 PhotonAnimatorViewEditor Class Reference	93
8.34.1 Member Function Documentation	93

8.34.1.1 OnInspectorGUI()	93
8.35 PhotonConverter Class Reference	93
8.35.1 Member Function Documentation	94
8.35.1.1 ConvertRpcAttribute(string path)	94
8.35.1.2 GetScriptsInFolder(string folder)	94
8.35.1.3 PickFolderAndConvertScripts()	94
8.35.1.4 RunConversion()	94
8.36 PhotonEditor Class Reference	94
8.36.1 Constructor & Destructor Documentation	95
8.36.1.1 PhotonEditor()	95
8.36.2 Member Function Documentation	95
8.36.2.1 ClearRpcList()	95
8.36.2.2 GetAllSubTypesInScripts(System.Type aBaseClass)	95
8.36.2.3 MenuItemHighlightSettings()	95
8.36.2.4 MenuItemOpenWizard()	95
8.36.2.5 OnGUI()	95
8.36.2.6 RegisterWithEmail(string email)	95
8.36.2.7 ShowRegistrationWizard()	95
8.36.2.8 UiMainWizard()	96
8.36.2.9 UiSetupApp()	96
8.36.2.10 Update()	96
8.36.2.11 UpdateRpcList()	96
8.36.3 Member Data Documentation	96
8.36.3.1 CurrentLang	96
8.36.3.2 DocumentationLocation	96
8.36.3.3 RegisterOrigin	96
8.36.3.4 scrollPos	96
8.36.3.5 UriAccountPage	96
8.36.3.6 UriAppIDExplained	96
8.36.3.7 UriCloudDashboard	96

8.36.3.8	UrlCompare	96
8.36.3.9	UrlDevNet	96
8.36.3.10	UrlForum	96
8.36.3.11	UrlFreeLicense	96
8.36.3.12	UrlHowToSetup	96
8.36.3.13	WindowType	96
8.37	PhotonGUI Class Reference	97
8.37.1	Member Function Documentation	97
8.37.1.1	AddButton()	97
8.37.1.2	ContainerBody(float height)	97
8.37.1.3	ContainerHeader(string headline)	97
8.37.1.4	ContainerHeaderFoldout(string headline, bool foldout)	97
8.37.1.5	ContainerHeaderToggle(string headline, bool toggle)	97
8.37.1.6	DrawGizmoOptions(Rect position, string label, SerializedProperty gizmo← EnabledProperty, SerializedProperty gizmoColorProperty, SerializedProperty gizmoTypeProperty, SerializedProperty gizmoSizeProperty)	97
8.37.1.7	DrawSplitter(Rect position)	97
8.37.2	Property Documentation	97
8.37.2.1	DefaultAddButtonStyle	97
8.37.2.2	DefaultContainerRowStyle	98
8.37.2.3	DefaultContainerStyle	98
8.37.2.4	DefaultRemoveButtonStyle	98
8.37.2.5	DefaultTitleStyle	98
8.37.2.6	FoldoutBold	98
8.37.2.7	HelpIcon	98
8.37.2.8	RichLabel	98
8.38	PhotonLagSimulationGui Class Reference	98
8.38.1	Detailed Description	99
8.38.2	Member Function Documentation	99
8.38.2.1	OnGUI()	99
8.38.2.2	Start()	99

8.38.3	Member Data Documentation	99
8.38.3.1	Visible	99
8.38.3.2	WindowId	99
8.38.3.3	WindowRect	99
8.38.4	Property Documentation	99
8.38.4.1	Peer	99
8.39	PhotonMessageInfo Class Reference	99
8.39.1	Detailed Description	100
8.39.2	Constructor & Destructor Documentation	100
8.39.2.1	PhotonMessageInfo()	100
8.39.2.2	PhotonMessageInfo(PhotonPlayer player, int timestamp, PhotonView view)	100
8.39.3	Member Function Documentation	100
8.39.3.1	ToString()	100
8.39.4	Member Data Documentation	100
8.39.4.1	photonView	100
8.39.4.2	sender	100
8.39.5	Property Documentation	100
8.39.5.1	timestamp	100
8.40	PhotonNetwork Class Reference	101
8.40.1	Detailed Description	106
8.40.2	Member Function Documentation	106
8.40.2.1	AllocateSceneViewID()	106
8.40.2.2	AllocateViewID()	106
8.40.2.3	CacheSendMonoMessageTargets(Type type)	106
8.40.2.4	CloseConnection(PhotonPlayer kickPlayer)	107
8.40.2.5	ConnectToBestCloudServer(string gameVersion)	107
8.40.2.6	ConnectToMaster(string masterServerAddress, int port, string applID, string gameVersion)	108
8.40.2.7	ConnectToRegion(CloudRegionCode region, string gameVersion)	108
8.40.2.8	ConnectUsingSettings(string gameVersion)	108
8.40.2.9	CreateRoom(string roomName)	109

8.40.2.10 CreateRoom(string roomName, RoomOptions roomOptions, TypedLobby typed↵ Lobby)	109
8.40.2.11 Destroy(PhotonView targetView)	110
8.40.2.12 Destroy(GameObject targetGo)	111
8.40.2.13 DestroyAll()	111
8.40.2.14 DestroyPlayerObjects(PhotonPlayer targetPlayer)	112
8.40.2.15 DestroyPlayerObjects(int targetPlayerId)	112
8.40.2.16 Disconnect()	112
8.40.2.17 EventCallback(byte eventCode, object content, int senderId)	112
8.40.2.18 FetchServerTimestamp()	113
8.40.2.19 FindFriends(string[] friendsToFind)	113
8.40.2.20 FindGameObjectsWithComponent(Type type)	113
8.40.2.21 GetPing()	114
8.40.2.22 GetRoomList()	114
8.40.2.23 InitializeSecurity()	114
8.40.2.24 Instantiate(string prefabName, Vector3 position, Quaternion rotation, int group)	114
8.40.2.25 Instantiate(string prefabName, Vector3 position, Quaternion rotation, int group, object[] data)	115
8.40.2.26 InstantiateSceneObject(string prefabName, Vector3 position, Quaternion rotation, int group, object[] data)	115
8.40.2.27 JoinLobby()	116
8.40.2.28 JoinLobby(TypedLobby typedLobby)	116
8.40.2.29 JoinOrCreateRoom(string roomName, RoomOptions roomOptions, TypedLobby typedLobby)	117
8.40.2.30 JoinRandomRoom()	117
8.40.2.31 JoinRandomRoom(Hashtable expectedCustomRoomProperties, byte expected↵ MaxPlayers)	117
8.40.2.32 JoinRandomRoom(Hashtable expectedCustomRoomProperties, byte expected↵ MaxPlayers, MatchmakingMode matchingType, TypedLobby typedLobby, string sqlLobbyFilter)	118
8.40.2.33 JoinRoom(string roomName)	118
8.40.2.34 LeaveLobby()	119
8.40.2.35 LeaveRoom()	119

8.40.2.36 LoadLevel(int levelNumber)	119
8.40.2.37 LoadLevel(string levelName)	119
8.40.2.38 NetworkStatisticsReset()	120
8.40.2.39 NetworkStatisticsToString()	120
8.40.2.40 OverrideBestCloudServer(CloudRegionCode region)	120
8.40.2.41 RaiseEvent(byte eventCode, object eventContent, bool sendReliable, Raise← EventOptions options)	120
8.40.2.42 RefreshCloudServerRating()	121
8.40.2.43 RemovePlayerCustomProperties(string[] customPropertiesToDelete)	121
8.40.2.44 RemoveRPCs(PhotonPlayer targetPlayer)	122
8.40.2.45 RemoveRPCs(PhotonView targetPhotonView)	122
8.40.2.46 RemoveRPCsInGroup(int targetGroup)	122
8.40.2.47 SendOutgoingCommands()	123
8.40.2.48 SetLevelPrefix(short prefix)	123
8.40.2.49 SetMasterClient(PhotonPlayer masterClientPlayer)	123
8.40.2.50 SetPlayerCustomProperties(Hashtable customProperties)	124
8.40.2.51 SetReceivingEnabled(int group, bool enabled)	124
8.40.2.52 SetReceivingEnabled(int[] enableGroups, int[] disableGroups)	125
8.40.2.53 SetSendingEnabled(int group, bool enabled)	125
8.40.2.54 SetSendingEnabled(int[] enableGroups, int[] disableGroups)	125
8.40.2.55 SwitchToProtocol(ConnectionProtocol cp)	125
8.40.2.56 UnAllocateViewID(int viewID)	126
8.40.2.57 WebRpc(string name, object parameters)	126
8.40.3 Member Data Documentation	127
8.40.3.1 BackgroundTimeout	127
8.40.3.2 InstantiateInRoomOnly	127
8.40.3.3 logLevel	127
8.40.3.4 MAX_VIEW_IDS	127
8.40.3.5 maxConnections	127
8.40.3.6 OnEventCall	127
8.40.3.7 PhotonServerSettings	128

8.40.3.8	precisionForFloatSynchronization	128
8.40.3.9	precisionForQuaternionSynchronization	128
8.40.3.10	precisionForVectorSynchronization	128
8.40.3.11	PrefabCache	128
8.40.3.12	SendMonoMessageTargets	128
8.40.3.13	SendMonoMessageTargetType	129
8.40.3.14	UsePrefabCache	129
8.40.3.15	UseRpcMonoBehaviourCache	129
8.40.3.16	versionPUN	129
8.40.4	Property Documentation	129
8.40.4.1	AuthValues	129
8.40.4.2	autoCleanUpPlayerObjects	129
8.40.4.3	autoJoinLobby	130
8.40.4.4	automaticallySyncScene	130
8.40.4.5	connected	130
8.40.4.6	connectedAndReady	130
8.40.4.7	connecting	130
8.40.4.8	connectionState	130
8.40.4.9	connectionStateDetailed	131
8.40.4.10	countOfPlayers	131
8.40.4.11	countOfPlayersInRooms	131
8.40.4.12	countOfPlayersOnMaster	131
8.40.4.13	countOfRooms	131
8.40.4.14	CrcCheckEnabled	131
8.40.4.15	EnableLobbyStatistics	131
8.40.4.16	Friends	132
8.40.4.17	FriendsListAge	132
8.40.4.18	gameVersion	132
8.40.4.19	inRoom	132
8.40.4.20	insideLobby	132

8.40.4.21 isMasterClient	132
8.40.4.22 isMessageQueueRunning	133
8.40.4.23 isNonMasterClientInRoom	133
8.40.4.24 lobby	133
8.40.4.25 LobbyStatistics	133
8.40.4.26 masterClient	133
8.40.4.27 MaxResendsBeforeDisconnect	134
8.40.4.28 NetworkStatisticsEnabled	134
8.40.4.29 offlineMode	134
8.40.4.30 otherPlayers	134
8.40.4.31 PacketLossByCrcCheck	134
8.40.4.32 player	134
8.40.4.33 playerList	135
8.40.4.34 playerName	135
8.40.4.35 PrefabPool	135
8.40.4.36 QuickResends	135
8.40.4.37 ResentReliableCommands	135
8.40.4.38 room	136
8.40.4.39 sendRate	136
8.40.4.40 sendRateOnSerialize	136
8.40.4.41 Server	136
8.40.4.42 ServerAddress	136
8.40.4.43 ServerTimestamp	136
8.40.4.44 time	137
8.40.4.45 unreliableCommandsLimit	137
8.41 PhotonPingManager Class Reference	137
8.41.1 Member Function Documentation	138
8.41.1.1 PingSocket(Region region)	138
8.41.1.2 ResolveHost(string hostName)	138
8.41.2 Member Data Documentation	138

8.41.2.1 Attempts	138
8.41.2.2 IgnoreInitialAttempt	138
8.41.2.3 MaxMilliseconsPerPing	138
8.41.2.4 UseNative	138
8.41.3 Property Documentation	138
8.41.3.1 BestRegion	138
8.41.3.2 Done	138
8.42 PhotonPlayer Class Reference	139
8.42.1 Detailed Description	140
8.42.2 Constructor & Destructor Documentation	140
8.42.2.1 PhotonPlayer(bool isLocal, int actorID, string name)	140
8.42.2.2 PhotonPlayer(bool isLocal, int actorID, Hashtable properties)	140
8.42.3 Member Function Documentation	140
8.42.3.1 Equals(object p)	140
8.42.3.2 Find(int ID)	140
8.42.3.3 Get(int id)	140
8.42.3.4 GetHashCode()	141
8.42.3.5 GetNext()	141
8.42.3.6 GetNextFor(PhotonPlayer currentPlayer)	141
8.42.3.7 GetNextFor(int currentPlayerId)	141
8.42.3.8 SetCustomProperties(Hashtable propertiesToSet, Hashtable expected↔ Values=null, bool webForward=false)	141
8.42.3.9 ToString()	142
8.42.3.10 ToStringFull()	142
8.42.4 Member Data Documentation	142
8.42.4.1 isLocal	142
8.42.4.2 TagObject	142
8.42.5 Property Documentation	142
8.42.5.1 allProperties	142
8.42.5.2 customProperties	142
8.42.5.3 ID	142

8.42.5.4	isMasterClient	143
8.42.5.5	name	143
8.43	PhotonRigidbody2DView Class Reference	143
8.43.1	Detailed Description	143
8.44	PhotonRigidbody2DViewEditor Class Reference	143
8.44.1	Member Function Documentation	144
8.44.1.1	OnInspectorGUI()	144
8.45	PhotonRigidbodyView Class Reference	144
8.45.1	Detailed Description	144
8.46	PhotonRigidbodyViewEditor Class Reference	144
8.46.1	Member Function Documentation	144
8.46.1.1	OnInspectorGUI()	144
8.47	PhotonStatsGui Class Reference	145
8.47.1	Detailed Description	145
8.47.2	Member Function Documentation	145
8.47.2.1	OnGUI()	145
8.47.2.2	Start()	145
8.47.2.3	TrafficStatsWindow(int windowID)	145
8.47.2.4	Update()	145
8.47.3	Member Data Documentation	146
8.47.3.1	buttonsOn	146
8.47.3.2	healthStatsVisible	146
8.47.3.3	statsOn	146
8.47.3.4	statsRect	146
8.47.3.5	statsWindowOn	146
8.47.3.6	trafficStatsOn	146
8.47.3.7	WindowId	146
8.48	PhotonStream Class Reference	146
8.48.1	Detailed Description	148
8.48.2	Constructor & Destructor Documentation	148

8.48.2.1	PhotonStream(bool write, object[] incomingData)	148
8.48.3	Member Function Documentation	148
8.48.3.1	PeekNext()	148
8.48.3.2	ReceiveNext()	148
8.48.3.3	SendNext(object obj)	148
8.48.3.4	Serialize(ref bool myBool)	148
8.48.3.5	Serialize(ref int myInt)	148
8.48.3.6	Serialize(ref string value)	149
8.48.3.7	Serialize(ref char value)	149
8.48.3.8	Serialize(ref short value)	149
8.48.3.9	Serialize(ref float obj)	149
8.48.3.10	Serialize(ref PhotonPlayer obj)	149
8.48.3.11	Serialize(ref Vector3 obj)	149
8.48.3.12	Serialize(ref Vector2 obj)	149
8.48.3.13	Serialize(ref Quaternion obj)	149
8.48.3.14	ToArray()	149
8.48.4	Property Documentation	149
8.48.4.1	Count	149
8.48.4.2	isReading	150
8.48.4.3	isWriting	150
8.49	PhotonStreamQueue Class Reference	150
8.49.1	Detailed Description	150
8.49.2	Constructor & Destructor Documentation	150
8.49.2.1	PhotonStreamQueue(int sampleRate)	150
8.49.3	Member Function Documentation	151
8.49.3.1	Deserialize(PhotonStream stream)	151
8.49.3.2	HasQueuedObjects()	151
8.49.3.3	ReceiveNext()	151
8.49.3.4	Reset()	151
8.49.3.5	SendNext(object obj)	151

8.49.3.6	Serialize(PhotonStream stream)	152
8.50	PhotonTransformView Class Reference	152
8.50.1	Detailed Description	152
8.50.2	Member Function Documentation	153
8.50.2.1	OnPhotonSerializeView(PhotonStream stream, PhotonMessageInfo info)	153
8.50.2.2	SetSynchronizedValues(Vector3 speed, float turnSpeed)	153
8.51	PhotonTransformViewEditor Class Reference	153
8.51.1	Member Function Documentation	154
8.51.1.1	OnEnable()	154
8.51.1.2	OnInspectorGUI()	154
8.52	PhotonTransformViewPositionControl Class Reference	154
8.52.1	Constructor & Destructor Documentation	154
8.52.1.1	PhotonTransformViewPositionControl(PhotonTransformViewPositionModel model)	154
8.52.2	Member Function Documentation	154
8.52.2.1	GetExtrapolatedPositionOffset()	154
8.52.2.2	GetNetworkPosition()	155
8.52.2.3	OnPhotonSerializeView(Vector3 currentPosition, PhotonStream stream, PhotonMessageInfo info)	155
8.52.2.4	SetSynchronizedValues(Vector3 speed, float turnSpeed)	155
8.52.2.5	UpdatePosition(Vector3 currentPosition)	155
8.53	PhotonTransformViewPositionModel Class Reference	155
8.53.1	Member Enumeration Documentation	156
8.53.1.1	ExtrapolateOptions	156
8.53.1.2	InterpolateOptions	156
8.53.2	Member Data Documentation	157
8.53.2.1	DrawErrorGizmo	157
8.53.2.2	ExtrapolateIncludingRoundTripTime	157
8.53.2.3	ExtrapolateNumberOfStoredPositions	157
8.53.2.4	ExtrapolateOption	157
8.53.2.5	ExtrapolateSpeed	157
8.53.2.6	InterpolateLerpSpeed	157

8.53.2.7	InterpolateMoveTowardsAcceleration	157
8.53.2.8	InterpolateMoveTowardsDeceleration	157
8.53.2.9	InterpolateMoveTowardsSpeed	157
8.53.2.10	InterpolateOption	157
8.53.2.11	InterpolateSpeedCurve	157
8.53.2.12	SynchronizeEnabled	157
8.53.2.13	TeleportEnabled	157
8.53.2.14	TeleportIfDistanceGreaterThan	157
8.54	PhotonTransformViewRotationControl Class Reference	158
8.54.1	Constructor & Destructor Documentation	158
8.54.1.1	PhotonTransformViewRotationControl(PhotonTransformViewRotationModel model)	158
8.54.2	Member Function Documentation	158
8.54.2.1	GetRotation(Quaternion currentRotation)	158
8.54.2.2	OnPhotonSerializeView(Quaternion currentRotation, PhotonStream stream, PhotonMessageInfo info)	158
8.55	PhotonTransformViewRotationModel Class Reference	158
8.55.1	Member Enumeration Documentation	158
8.55.1.1	InterpolateOptions	158
8.55.2	Member Data Documentation	159
8.55.2.1	InterpolateLerpSpeed	159
8.55.2.2	InterpolateOption	159
8.55.2.3	InterpolateRotateTowardsSpeed	159
8.55.2.4	SynchronizeEnabled	159
8.56	PhotonTransformViewScaleControl Class Reference	159
8.56.1	Constructor & Destructor Documentation	159
8.56.1.1	PhotonTransformViewScaleControl(PhotonTransformViewScaleModel model)	159
8.56.2	Member Function Documentation	159
8.56.2.1	GetScale(Vector3 currentScale)	159
8.56.2.2	OnPhotonSerializeView(Vector3 currentScale, PhotonStream stream, PhotonMessageInfo info)	159
8.57	PhotonTransformViewScaleModel Class Reference	159

8.57.1	Member Enumeration Documentation	160
8.57.1.1	InterpolateOptions	160
8.57.2	Member Data Documentation	160
8.57.2.1	InterpolateLerpSpeed	160
8.57.2.2	InterpolateMoveTowardsSpeed	160
8.57.2.3	InterpolateOption	160
8.57.2.4	SynchronizeEnabled	160
8.58	PhotonView Class Reference	160
8.58.1	Detailed Description	162
8.58.2	Member Function Documentation	162
8.58.2.1	DeserializeView(PhotonStream stream, PhotonMessageInfo info)	162
8.58.2.2	Find(int viewID)	162
8.58.2.3	Get(Component component)	162
8.58.2.4	Get(GameObject gameObj)	162
8.58.2.5	RefreshRpcMonoBehaviourCache()	162
8.58.2.6	RequestOwnership()	163
8.58.2.7	RPC(string methodName, PhotonTargets target, params object[] parameters)	163
8.58.2.8	RPC(string methodName, PhotonPlayer targetPlayer, params object[] parameters)	163
8.58.2.9	RpcSecure(string methodName, PhotonTargets target, bool encrypt, params object[] parameters)	164
8.58.2.10	RpcSecure(string methodName, PhotonPlayer targetPlayer, bool encrypt, params object[] parameters)	164
8.58.2.11	SerializeView(PhotonStream stream, PhotonMessageInfo info)	164
8.58.2.12	ToString()	164
8.58.2.13	TransferOwnership(PhotonPlayer newOwner)	164
8.58.2.14	TransferOwnership(int newOwnerId)	165
8.58.3	Member Data Documentation	165
8.58.3.1	group	165
8.58.3.2	instantiationId	165
8.58.3.3	observed	165
8.58.3.4	ObservedComponents	165

8.58.3.5	onSerializeRigidBodyOption	165
8.58.3.6	onSerializeTransformOption	165
8.58.3.7	ownerId	165
8.58.3.8	ownershipTransfer	165
8.58.3.9	prefixBackup	165
8.58.3.10	synchronization	165
8.58.4	Property Documentation	165
8.58.4.1	CreatorActorNr	165
8.58.4.2	instantiationData	165
8.58.4.3	isMine	166
8.58.4.4	isOwnerActive	166
8.58.4.5	isSceneView	166
8.58.4.6	owner	166
8.58.4.7	OwnerActorNr	166
8.58.4.8	prefix	166
8.58.4.9	viewID	166
8.59	PhotonViewHandler Class Reference	166
8.59.1	Member Function Documentation	167
8.59.1.1	GetID(int idOffset, HashSet< int > usedInstanceViewNumbers)	167
8.59.1.2	LoadAllScenesToFix()	167
8.60	PhotonViewInspector Class Reference	167
8.60.1	Member Function Documentation	167
8.60.1.1	OnInspectorGUI()	167
8.61	PickupItem Class Reference	167
8.61.1	Detailed Description	168
8.61.2	Member Function Documentation	168
8.61.2.1	Drop()	168
8.61.2.2	Drop(Vector3 newPosition)	169
8.61.2.3	OnPhotonSerializeView(PhotonStream stream, PhotonMessageInfo info)	169
8.61.2.4	OnTriggerEnter(Collider other)	169

8.61.2.5	Pickup()	169
8.61.2.6	PunPickup(PhotonMessageInfo msgInfo)	169
8.61.3	Member Data Documentation	169
8.61.3.1	DisabledPickupItems	169
8.61.3.2	OnPickedUpCall	169
8.61.3.3	PickupIsMine	170
8.61.3.4	PickupOnTrigger	170
8.61.3.5	SecondsBeforeRespawn	170
8.61.3.6	SentPickup	170
8.61.3.7	TimeOfRespawn	170
8.61.4	Property Documentation	170
8.61.4.1	ViewID	170
8.62	PickupItemSimple Class Reference	171
8.62.1	Detailed Description	171
8.62.2	Member Function Documentation	171
8.62.2.1	OnTriggerEnter(Collider other)	171
8.62.2.2	Pickup()	171
8.62.2.3	PunPickupSimple(PhotonMessageInfo msgInfo)	171
8.62.2.4	RespawnAfter()	171
8.62.3	Member Data Documentation	171
8.62.3.1	PickupOnCollide	171
8.62.3.2	SecondsBeforeRespawn	171
8.62.3.3	SentPickup	171
8.63	PickupItemSyncer Class Reference	172
8.63.1	Detailed Description	172
8.63.2	Member Function Documentation	172
8.63.2.1	AskForPickupItemSpawnTimes()	172
8.63.2.2	OnJoinedRoom()	172
8.63.2.3	OnPhotonPlayerConnected(PhotonPlayer newPlayer)	172
8.63.2.4	PickupItemInit(double timeBase, float[] inactivePickupsAndTimes)	172

8.63.2.5	RequestForPickupTimes(PhotonMessageInfo msgInfo)	172
8.63.3	Member Data Documentation	172
8.63.3.1	IsWaitingForPickupInit	172
8.64	PingMonoEditor Class Reference	173
8.64.1	Detailed Description	173
8.64.2	Member Function Documentation	173
8.64.2.1	Dispose()	173
8.64.2.2	Done()	173
8.64.2.3	StartPing(string ip)	173
8.65	PointedAtGameObjectInfo Class Reference	173
8.66	Photon.PunBehaviour Class Reference	173
8.66.1	Detailed Description	175
8.66.2	Member Function Documentation	175
8.66.2.1	OnConnectedToMaster()	175
8.66.2.2	OnConnectedToPhoton()	175
8.66.2.3	OnConnectionFail(DisconnectCause cause)	176
8.66.2.4	OnCreatedRoom()	176
8.66.2.5	OnCustomAuthenticationFailed(string debugMessage)	176
8.66.2.6	OnDisconnectedFromPhoton()	176
8.66.2.7	OnFailedToConnectToPhoton(DisconnectCause cause)	177
8.66.2.8	OnJoinedLobby()	177
8.66.2.9	OnJoinedRoom()	177
8.66.2.10	OnLeftLobby()	177
8.66.2.11	OnLeftRoom()	178
8.66.2.12	OnLobbyStatisticsUpdate()	178
8.66.2.13	OnMasterClientSwitched(PhotonPlayer newMasterClient)	178
8.66.2.14	OnOwnershipRequest(object[] viewAndPlayer)	178
8.66.2.15	OnPhotonCreateRoomFailed(object[] codeAndMsg)	178
8.66.2.16	OnPhotonCustomRoomPropertiesChanged(Hashtable propertiesThatChanged)	180
8.66.2.17	OnPhotonInstantiate(PhotonMessageInfo info)	180

8.66.2.18 OnPhotonJoinRoomFailed(object[] codeAndMsg)	180
8.66.2.19 OnPhotonMaxCccuReached()	181
8.66.2.20 OnPhotonPlayerConnected(PhotonPlayer newPlayer)	181
8.66.2.21 OnPhotonPlayerDisconnected(PhotonPlayer otherPlayer)	181
8.66.2.22 OnPhotonPlayerPropertiesChanged(object[] playerAndUpdatedProps)	181
8.66.2.23 OnPhotonRandomJoinFailed(object[] codeAndMsg)	182
8.66.2.24 OnReceivedRoomListUpdate()	182
8.66.2.25 OnUpdatedFriendList()	182
8.66.2.26 OnWebRpcResponse(OperationResponse response)	183
8.67 PunPlayerScores Class Reference	183
8.67.1 Member Data Documentation	183
8.67.1.1 PlayerScoreProp	183
8.68 PunRPC Class Reference	183
8.68.1 Detailed Description	184
8.69 PunSceneSettings Class Reference	184
8.69.1 Member Function Documentation	184
8.69.1.1 MinViewIdForScene(string scene)	184
8.69.2 Member Data Documentation	184
8.69.2.1 MinViewIdPerScene	184
8.69.3 Property Documentation	184
8.69.3.1 Instance	184
8.69.3.2 PunSceneSettingsCsPath	184
8.70 PunTeams Class Reference	185
8.70.1 Detailed Description	185
8.70.2 Member Enumeration Documentation	185
8.70.2.1 Team	185
8.70.3 Member Function Documentation	186
8.70.3.1 OnJoinedRoom()	186
8.70.3.2 OnPhotonPlayerPropertiesChanged(object[] playerAndUpdatedProps)	186
8.70.3.3 Start()	186

8.70.3.4	UpdateTeams()	186
8.70.4	Member Data Documentation	186
8.70.4.1	PlayersPerTeam	186
8.70.4.2	TeamPlayerProp	186
8.71	PunWizardText Class Reference	187
8.71.1	Member Data Documentation	188
8.71.1.1	AlreadyRegisteredInfo	188
8.71.1.2	AppliedToSettingsInfo	188
8.71.1.3	CancelButton	188
8.71.1.4	CloseWindowButton	188
8.71.1.5	ComparisonPageButton	188
8.71.1.6	ConnectionInfo	188
8.71.1.7	ConnectionTitle	188
8.71.1.8	ConverterLabel	188
8.71.1.9	DocumentationLabel	188
8.71.1.10	EmailOrAppldLabel	188
8.71.1.11	ErrorTextTitle	189
8.71.1.12	FullRPCListLabel	189
8.71.1.13	FullRPCListTitle	189
8.71.1.14	IncorrectRPCListLabel	189
8.71.1.15	IncorrectRPCListTitle	189
8.71.1.16	LocateSettingsButton	189
8.71.1.17	MainMenuButton	189
8.71.1.18	MobileExportNoteLabel	189
8.71.1.19	MobilePunPlusExportNoteLabel	189
8.71.1.20	OkButton	189
8.71.1.21	OpenCloudDashboardText	189
8.71.1.22	OpenCloudDashboardTooltip	189
8.71.1.23	OpenDevNetText	189
8.71.1.24	OpenDevNetTooltip	189

8.71.1.25 OpenForumText	189
8.71.1.26 OpenForumTooltip	189
8.71.1.27 OpenPDFText	189
8.71.1.28 OpenPDFTooltip	189
8.71.1.29 OwnHostCloudCompareLabel	189
8.71.1.30 PUNNameReplaceLabel	189
8.71.1.31 PUNNameReplaceTitle	190
8.71.1.32 PUNWizardLabel	190
8.71.1.33 RegisteredNewAccountInfo	190
8.71.1.34 RemoveOutdatedRPCsLabel	190
8.71.1.35 RpcFoundDialogTitle	190
8.71.1.36 RpcFoundMessage	190
8.71.1.37 RPCListCleared	190
8.71.1.38 RpcReplaceButton	190
8.71.1.39 RpcSkipReplace	190
8.71.1.40 ServerSettingsCleanedWarning	190
8.71.1.41 SettingsButton	190
8.71.1.42 SettingsHighlightLabel	190
8.71.1.43 SetupButton	190
8.71.1.44 SetupCompleteInfo	190
8.71.1.45 SetupServerCloudLabel	190
8.71.1.46 SetupWizardInfo	190
8.71.1.47 SetupWizardTitle	190
8.71.1.48 SetupWizardWarningMessage	190
8.71.1.49 SetupWizardWarningTitle	190
8.71.1.50 SkipButton	190
8.71.1.51 SkipRegistrationInfo	191
8.71.1.52 SkipRPCListUpdateLabel	191
8.71.1.53 StartButton	191
8.71.1.54 UNtoPUNLabel	191

8.71.1.55 WarningPhotonDisconnect	191
8.71.1.56 WindowTitle	191
8.71.1.57 WizardMainWindowInfo	191
8.72 QuitOnEscapeOrBack Class Reference	191
8.73 RaiseEventOptions Class Reference	191
8.73.1 Detailed Description	192
8.73.2 Member Data Documentation	192
8.73.2.1 CachingOption	192
8.73.2.2 Default	192
8.73.2.3 Encrypt	192
8.73.2.4 ForwardToWebhook	192
8.73.2.5 InterestGroup	192
8.73.2.6 Receivers	192
8.73.2.7 SequenceChannel	192
8.73.2.8 TargetActors	193
8.74 Region Class Reference	193
8.74.1 Member Function Documentation	193
8.74.1.1 Parse(string codeAsString)	193
8.74.1.2 ToString()	193
8.74.2 Member Data Documentation	193
8.74.2.1 Code	193
8.74.2.2 HostAndPort	193
8.74.2.3 Ping	193
8.75 Room Class Reference	194
8.75.1 Detailed Description	194
8.75.2 Member Function Documentation	195
8.75.2.1 SetCustomProperties(Hashtable propertiesToSet, Hashtable expected↔ Values=null, bool webForward=false)	195
8.75.2.2 SetPropertiesListedInLobby(string[] propsListedInLobby)	195
8.75.2.3 ToString()	196
8.75.2.4 ToStringFull()	196

8.75.3	Property Documentation	196
8.75.3.1	autoCleanUp	196
8.75.3.2	maxPlayers	196
8.75.3.3	name	196
8.75.3.4	open	196
8.75.3.5	playerCount	197
8.75.3.6	propertiesListedInLobby	197
8.75.3.7	visible	197
8.76	RoomInfo Class Reference	197
8.76.1	Detailed Description	198
8.76.2	Member Function Documentation	198
8.76.2.1	Equals(object p)	198
8.76.2.2	GetHashCode()	198
8.76.2.3	ToString()	198
8.76.2.4	ToStringFull()	199
8.76.3	Member Data Documentation	199
8.76.3.1	autoCleanUpField	199
8.76.3.2	maxPlayersField	199
8.76.3.3	nameField	199
8.76.3.4	openField	199
8.76.3.5	visibleField	199
8.76.4	Property Documentation	199
8.76.4.1	customProperties	199
8.76.4.2	isLocalClientInside	200
8.76.4.3	maxPlayers	200
8.76.4.4	name	200
8.76.4.5	open	200
8.76.4.6	playerCount	200
8.76.4.7	removedFromList	200
8.76.4.8	visible	200

8.77 RoomOptions Class Reference	201
8.77.1 Detailed Description	201
8.77.2 Member Data Documentation	201
8.77.2.1 customRoomProperties	201
8.77.2.2 customRoomPropertiesForLobby	201
8.77.2.3 maxPlayers	202
8.77.3 Property Documentation	202
8.77.3.1 cleanupCacheOnLeave	202
8.77.3.2 isOpen	202
8.77.3.3 isVisible	202
8.77.3.4 suppressRoomEvents	202
8.78 UnityEngine.SceneManagement.SceneManager Class Reference	203
8.78.1 Detailed Description	203
8.78.2 Member Function Documentation	203
8.78.2.1 LoadScene(string name)	203
8.78.2.2 LoadScene(int buildIndex)	203
8.79 SceneManagerHelper Class Reference	203
8.79.1 Property Documentation	203
8.79.1.1 ActiveSceneBuildIndex	203
8.79.1.2 ActiveSceneName	203
8.80 SceneSetting Class Reference	204
8.80.1 Member Data Documentation	204
8.80.1.1 minViewId	204
8.80.1.2 sceneName	204
8.81 ScoreExtensions Class Reference	204
8.81.1 Member Function Documentation	204
8.81.1.1 AddScore(this PhotonPlayer player, int scoreToAddToCurrent)	204
8.81.1.2 GetScore(this PhotonPlayer player)	204
8.81.1.3 SetScore(this PhotonPlayer player, int newScore)	204
8.82 ServerSettings Class Reference	204

8.82.1 Detailed Description	205
8.82.2 Member Enumeration Documentation	205
8.82.2.1 HostingOption	205
8.82.3 Member Function Documentation	206
8.82.3.1 ToString()	206
8.82.3.2 UseCloud(string cloudAppid)	206
8.82.3.3 UseCloud(string cloudAppid, CloudRegionCode code)	206
8.82.3.4 UseCloudBestRegion(string cloudAppid)	206
8.82.3.5 UseMyServer(string serverAddress, int serverPort, string application)	206
8.82.4 Member Data Documentation	206
8.82.4.1 AppID	206
8.82.4.2 DisableAutoOpenWizard	206
8.82.4.3 EnabledRegions	206
8.82.4.4 EnableLobbyStatistics	206
8.82.4.5 HostType	206
8.82.4.6 JoinLobby	206
8.82.4.7 PreferredRegion	206
8.82.4.8 Protocol	206
8.82.4.9 RpcList	206
8.82.4.10 ServerAddress	206
8.82.4.11 ServerPort	206
8.83 ServerSettingsInspector Class Reference	206
8.83.1 Member Enumeration Documentation	207
8.83.1.1 ProtocolChoices	207
8.83.2 Member Function Documentation	207
8.83.2.1 IsAppId(string val)	207
8.83.2.2 OnInspectorGUI()	207
8.84 ServerTime Class Reference	208
8.85 ShowInfoOfPlayer Class Reference	208
8.85.1 Detailed Description	208

8.85.2	Member Data Documentation	208
8.85.2.1	CharacterSize	208
8.85.2.2	DisableOnOwnObjects	208
8.85.2.3	font	208
8.86	ShowStatusWhenConnecting Class Reference	208
8.86.1	Member Data Documentation	209
8.86.1.1	Skin	209
8.87	SmoothSyncMovement Class Reference	209
8.87.1	Member Function Documentation	209
8.87.1.1	Awake()	209
8.87.1.2	OnPhotonSerializeView(PhotonStream stream, PhotonMessageInfo info)	209
8.87.1.3	Update()	209
8.87.2	Member Data Documentation	209
8.87.2.1	SmoothingDelay	209
8.88	SupportLogger Class Reference	210
8.88.1	Member Function Documentation	210
8.88.1.1	Start()	210
8.88.2	Member Data Documentation	210
8.88.2.1	LogTrafficStats	210
8.89	SupportLogging Class Reference	210
8.89.1	Member Function Documentation	211
8.89.1.1	LogStats()	211
8.89.1.2	OnApplicationPause(bool pause)	211
8.89.1.3	OnApplicationQuit()	211
8.89.1.4	OnConnectedToPhoton()	211
8.89.1.5	OnCreatedRoom()	211
8.89.1.6	OnDisconnectedFromPhoton()	211
8.89.1.7	OnFailedToConnectToPhoton(DisconnectCause cause)	211
8.89.1.8	OnJoinedLobby()	211
8.89.1.9	OnJoinedRoom()	211

8.89.1.10 OnLeftRoom()	211
8.89.1.11 Start()	211
8.89.2 Member Data Documentation	211
8.89.2.1 LogTrafficStats	211
8.90 PhotonAnimatorView.SynchronizedLayer Class Reference	211
8.90.1 Member Data Documentation	212
8.90.1.1 LayerIndex	212
8.90.1.2 SynchronizeType	212
8.91 PhotonAnimatorView.SynchronizedParameter Class Reference	212
8.91.1 Member Data Documentation	212
8.91.1.1 Name	212
8.91.1.2 SynchronizeType	212
8.91.1.3 Type	212
8.92 TeamExtensions Class Reference	212
8.92.1 Detailed Description	213
8.92.2 Member Function Documentation	213
8.92.2.1 GetTeam(this PhotonPlayer player)	213
8.92.2.2 SetTeam(this PhotonPlayer player, PunTeams.Team team)	213
8.93 ExitGames.Client.DemoParticle.TimeKeeper Class Reference	213
8.93.1 Detailed Description	214
8.93.2 Constructor & Destructor Documentation	214
8.93.2.1 TimeKeeper(int interval)	214
8.93.3 Member Function Documentation	214
8.93.3.1 Reset()	214
8.93.4 Property Documentation	214
8.93.4.1 Interval	214
8.93.4.2 IsEnabled	215
8.93.4.3 ShouldExecute	215
8.94 TypedLobby Class Reference	215
8.94.1 Detailed Description	216

8.94.2	Constructor & Destructor Documentation	216
8.94.2.1	TypedLobby()	216
8.94.2.2	TypedLobby(string name, LobbyType type)	216
8.94.3	Member Function Documentation	216
8.94.3.1	ToString()	216
8.94.4	Member Data Documentation	216
8.94.4.1	Default	216
8.94.4.2	Name	216
8.94.4.3	Type	216
8.94.5	Property Documentation	216
8.94.5.1	IsDefault	216
8.95	TypedLobbyInfo Class Reference	216
8.95.1	Member Function Documentation	217
8.95.1.1	ToString()	217
8.95.2	Member Data Documentation	217
8.95.2.1	PlayerCount	217
8.95.2.2	RoomCount	217
8.96	WebRpcResponse Class Reference	217
8.96.1	Detailed Description	218
8.96.2	Constructor & Destructor Documentation	218
8.96.2.1	WebRpcResponse(OperationResponse response)	218
8.96.3	Member Function Documentation	218
8.96.3.1	ToStringFull()	218
8.96.4	Property Documentation	218
8.96.4.1	DebugMessage	218
8.96.4.2	Name	218
8.96.4.3	Parameters	218
8.96.4.4	ReturnCode	218

9	File Documentation	219
9.1	C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Editor/↔ PhotonNetwork/AccountService.cs File Reference	219
9.2	C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Editor/↔ PhotonNetwork/PhotonConverter.cs File Reference	219
9.3	C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Editor/↔ PhotonNetwork/PhotonEditor.cs File Reference	219
9.4	C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Editor/↔ PhotonNetwork/PhotonGUI.cs File Reference	219
9.5	C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Editor/↔ PhotonNetwork/PhotonViewHandler.cs File Reference	220
9.5.1	Typedef Documentation	220
9.5.1.1	Debug	220
9.6	C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Editor/↔ PhotonNetwork/PhotonViewInspector.cs File Reference	220
9.7	C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Editor/↔ PhotonNetwork/PhotonViewPrefabApply.cs File Reference	220
9.8	C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Editor/↔ PhotonNetwork/PunSceneSettings.cs File Reference	220
9.9	C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Editor/↔ PhotonNetwork/ReorderableListResources.cs File Reference	220
9.10	C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Editor/↔ PhotonNetwork/ServerSettingsInspector.cs File Reference	221
9.11	C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Editor/↔ PhotonNetwork/Views/PhotonAnimatorViewEditor.cs File Reference	221
9.12	C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Editor/↔ PhotonNetwork/Views/PhotonRigidbody2DViewEditor.cs File Reference	221
9.13	C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Editor/↔ PhotonNetwork/Views/PhotonRigidbodyViewEditor.cs File Reference	221
9.14	C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Editor/↔ PhotonNetwork/Views/PhotonTransformViewEditor.cs File Reference	221
9.15	C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/↔ PhotonNetwork/CustomTypes.cs File Reference	222
9.15.1	Detailed Description	222
9.16	C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/↔ PhotonNetwork/Enums.cs File Reference	222
9.16.1	Detailed Description	223
9.16.2	Enumeration Type Documentation	223

9.16.2.1	CloudRegionCode	223
9.16.2.2	CloudRegionFlag	224
9.16.2.3	ConnectionState	224
9.16.2.4	ServerConnection	224
9.17	C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/↔ PhotonNetwork/Extensions.cs File Reference	225
9.17.1	Typedef Documentation	225
9.17.1.1	Hashtable	225
9.17.1.2	SupportClass	225
9.18	C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/↔ PhotonNetwork/FriendInfo.cs File Reference	225
9.19	C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/↔ PhotonNetwork/GizmoType.cs File Reference	225
9.20	C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/↔ PhotonNetwork/LoadbalancingPeer.cs File Reference	226
9.20.1	Typedef Documentation	227
9.20.1.1	Hashtable	227
9.20.2	Enumeration Type Documentation	227
9.20.2.1	CustomAuthenticationType	227
9.20.2.2	LobbyType	228
9.21	C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/↔ PhotonNetwork/NetworkingPeer.cs File Reference	228
9.21.1	Typedef Documentation	228
9.21.1.1	Hashtable	228
9.22	C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/↔ PhotonNetwork/PhotonClasses.cs File Reference	228
9.22.1	Detailed Description	229
9.22.2	Typedef Documentation	230
9.22.2.1	Hashtable	230
9.23	C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/↔ PhotonNetwork/PhotonHandler.cs File Reference	230
9.23.1	Typedef Documentation	230
9.23.1.1	Debug	230

9.23.1.2	Hashtable	230
9.24	C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/↔ PhotonNetwork/PhotonLagSimulationGui.cs File Reference	230
9.24.1	Detailed Description	230
9.25	C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/↔ PhotonNetwork/PhotonNetwork.cs File Reference	230
9.25.1	Typedef Documentation	231
9.25.1.1	Debug	231
9.25.1.2	Hashtable	231
9.26	C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/↔ PhotonNetwork/PhotonPlayer.cs File Reference	231
9.26.1	Typedef Documentation	231
9.26.1.1	Hashtable	231
9.27	C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/↔ PhotonNetwork/PhotonStatsGui.cs File Reference	231
9.27.1	Detailed Description	231
9.28	C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/↔ PhotonNetwork/PhotonStreamQueue.cs File Reference	232
9.29	C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/↔ PhotonNetwork/PhotonView.cs File Reference	232
9.29.1	Enumeration Type Documentation	232
9.29.1.1	OnSerializeRigidBody	232
9.29.1.2	OnSerializeTransform	232
9.29.1.3	OwnershipOption	233
9.29.1.4	ViewSynchronization	233
9.30	C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/↔ PhotonNetwork/PingCloudRegions.cs File Reference	233
9.30.1	Typedef Documentation	233
9.30.1.1	Debug	233
9.31	C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/↔ PhotonNetwork/Room.cs File Reference	233
9.32	C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/↔ PhotonNetwork/RoomInfo.cs File Reference	234
9.33	C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/↔ PhotonNetwork/RPC.cs File Reference	234

9.33.1 Detailed Description	234
9.34 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/↔ PhotonNetwork/RpclIndexComponent.cs File Reference	234
9.34.1 Detailed Description	234
9.35 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/↔ PhotonNetwork/ServerSettings.cs File Reference	234
9.35.1 Detailed Description	235
9.36 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/↔ PhotonNetwork/SocketUdp.cs File Reference	235
9.37 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/↔ PhotonNetwork/SocketWebTcp.cs File Reference	235
9.38 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/↔ PhotonNetwork/Views/PhotonAnimatorView.cs File Reference	235
9.39 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/↔ PhotonNetwork/Views/PhotonRigidbody2DView.cs File Reference	235
9.40 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/↔ PhotonNetwork/Views/PhotonRigidbodyView.cs File Reference	236
9.41 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/↔ PhotonNetwork/Views/PhotonTransformView.cs File Reference	236
9.42 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/↔ PhotonNetwork/Views/PhotonTransformViewPositionControl.cs File Reference	236
9.43 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/↔ PhotonNetwork/Views/PhotonTransformViewPositionModel.cs File Reference	236
9.44 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/↔ PhotonNetwork/Views/PhotonTransformViewRotationControl.cs File Reference	236
9.45 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/↔ PhotonNetwork/Views/PhotonTransformViewRotationModel.cs File Reference	237
9.46 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/↔ PhotonNetwork/Views/PhotonTransformViewScaleControl.cs File Reference	237
9.47 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/↔ PhotonNetwork/Views/PhotonTransformViewScaleModel.cs File Reference	237
9.48 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Utility↔ Scripts/ConnectAndJoinRandom.cs File Reference	237
9.49 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Utility↔ Scripts/HighlightOwnedGameObj.cs File Reference	237
9.50 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Utility↔ Scripts/InputToEvent.cs File Reference	238
9.51 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Utility↔ Scripts/InRoomChat.cs File Reference	238

9.52	C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Utility↔ Scripts/InRoomRoundTimer.cs File Reference	238
9.53	C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Utility↔ Scripts/ManualPhotonViewAllocator.cs File Reference	238
9.54	C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Utility↔ Scripts/MoveByKeys.cs File Reference	238
9.55	C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Utility↔ Scripts/OnAwakeUsePhotonView.cs File Reference	239
9.56	C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Utility↔ Scripts/OnClickDestroy.cs File Reference	239
9.57	C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Utility↔ Scripts/OnClickInstantiate.cs File Reference	239
9.58	C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Utility↔ Scripts/OnClickLoadSomething.cs File Reference	239
9.59	C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Utility↔ Scripts/OnJoinedInstantiate.cs File Reference	239
9.60	C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Utility↔ Scripts/OnStartDelete.cs File Reference	239
9.61	C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Utility↔ Scripts/PickupItem.cs File Reference	240
9.61.1	Typedef Documentation	240
9.61.1.1	Hashtable	240
9.62	C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Utility↔ Scripts/PickupItemSimple.cs File Reference	240
9.63	C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Utility↔ Scripts/PickupItemSyncer.cs File Reference	240
9.63.1	Typedef Documentation	240
9.63.1.1	Hashtable	240
9.64	C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Utility↔ Scripts/PointedAtGameObjectInfo.cs File Reference	240
9.65	C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Utility↔ Scripts/PunPlayerScores.cs File Reference	241
9.65.1	Typedef Documentation	241
9.65.1.1	Hashtable	241
9.66	C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Utility↔ Scripts/PunTeams.cs File Reference	241
9.66.1	Typedef Documentation	241

9.66.1.1	Hashtable	241
9.67	C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Utility↔ Scripts/QuitOnEscapeOrBack.cs File Reference	241
9.68	C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Utility↔ Scripts/ServerTime.cs File Reference	242
9.69	C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Utility↔ Scripts/ShowInfoOfPlayer.cs File Reference	242
9.70	C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Utility↔ Scripts/ShowStatusWhenConnecting.cs File Reference	242
9.71	C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Utility↔ Scripts/SmoothSyncMovement.cs File Reference	242
9.72	C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Utility↔ Scripts/SupportLogger.cs File Reference	242
9.73	C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Utility↔ Scripts/TimeKeeper.cs File Reference	242
Index		243

Chapter 1

Module Index

1.1 Modules

Here is a list of all modules:

Public API	17
Optional Gui Elements	26

Chapter 2

Namespace Index

2.1 Packages

Here are the packages with brief descriptions (if available):

ExitGames	27
ExitGames.Client	27
ExitGames.Client.DemoParticle	27
ExitGames.Client.GUI	27
ExitGames.Client.Photon	28
Photon	30
Rotorz	31
Rotorz.ReorderableList	31
Rotorz.ReorderableList.Internal	31
UnityEngine	31
UnityEngine.SceneManagement	31

Chapter 3

Hierarchical Index

3.1 Class Hierarchy

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

AccountService	33
ExitGames.Client.Photon.ActorProperties	35
Attribute	
HelpURL	53
PunRPC	183
AuthenticationValues	35
Editor	
PhotonAnimatorViewEditor	93
PhotonRigidbody2DViewEditor	143
PhotonRigidbodyViewEditor	144
PhotonTransformViewEditor	153
PhotonViewInspector	167
ServerSettingsInspector	206
EditorWindow	
PhotonEditor	94
PhotonViewHandler	166
ExitGames.Client.Photon.LoadbalancingPeer.EnterRoomParams	39
ExitGames.Client.Photon.ErrorCode	40
ExitGames.Client.Photon.EventCode	44
Extensions	46
FriendInfo	49
GameObjectExtensions	50
ExitGames.Client.Photon.GamePropertyKey	51
ExitGames.Client.GUI.GizmoTypeDrawer	53
IPunCallbacks	58
Photon.PunBehaviour	173
IPunObservable	67
PhotonTransformView	152
PickupItem	167
IPunPrefabPool	68
MonoBehaviour	
InputToEvent	54
InRoomRoundTimer	56
ManualPhotonViewAllocator	69
OnStartDelete	76

PhotonRigidbody2DView	143
PhotonStatsGui	145
PhotonTransformView	152
PunPlayerScores	183
QuitOnEscapeOrBack	191
ServerTime	208
SupportLogger	210
SupportLogging	210
MonoBehaviour	
OnClickInstantiate	74
OnClickLoadSomething	74
OnJoinedInstantiate	75
Photon.MonoBehaviour	70
ConnectAndJoinRandom	38
HighlightOwnedGameObj	54
InRoomChat	55
MoveByKeys	70
OnAwakeUsePhotonView	72
OnClickDestroy	73
Photon.PunBehaviour	173
PhotonConverter	93
PhotonView	160
PickupItem	167
PickupItemSimple	171
PickupItemSyncer	172
ShowInfoOfPlayer	208
SmoothSyncMovement	209
PhotonAnimatorView	90
PhotonLagSimulationGui	98
PhotonRigidbodyView	144
PointedAtGameObjectInfo	173
PunTeams	185
ShowStatusWhenConnecting	208
ExitGames.Client.Photon.OperationCode	76
ExitGames.Client.Photon.LoadbalancingPeer.OpJoinRandomRoomParams	79
ExitGames.Client.Photon.ParameterCode	80
PhotonGUI	97
PhotonMessageInfo	99
PhotonNetwork	101
PhotonPing	
PingMonoEditor	173
PhotonPingManager	137
PhotonPlayer	139
PhotonStream	146
PhotonStreamQueue	150
PhotonTransformViewPositionControl	154
PhotonTransformViewPositionModel	155
PhotonTransformViewRotationControl	158
PhotonTransformViewRotationModel	158
PhotonTransformViewScaleControl	159
PhotonTransformViewScaleModel	159
PunWizardText	187
RaiseEventOptions	191
Region	193
RoomInfo	197
Room	194
RoomOptions	201
UnityEngine.SceneManagement.SceneManager	203

SceneManagerHelper	203
SceneSetting	204
ScoreExtensions	204
ScriptableObject	
PunSceneSettings	184
ServerSettings	204
PhotonAnimatorView.SynchronizedLayer	211
PhotonAnimatorView.SynchronizedParameter	212
TeamExtensions	212
ExitGames.Client.DemoParticle.TimeKeeper	213
TypedLobby	215
TypedLobbyInfo	216
WebRpcResponse	217

Chapter 4

Class Index

4.1 Class List

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

AccountService	33
ExitGames.Client.Photon.ActorProperties Class for constants	35
AuthenticationValues Container for user authentication in Photon	35
ConnectAndJoinRandom This script automatically connects to Photon (using the settings file), tries to join a random room and creates one if none was found (which is ok)	38
ExitGames.Client.Photon.LoadbalancingPeer.EnterRoomParams	39
ExitGames.Client.Photon.ErrorCode ErrorCode defines the default codes associated with Photon client/server communication . . .	40
ExitGames.Client.Photon.EventCode Class for constants	44
Extensions This static class defines some useful extension methods for several existing classes (e.g . . .	46
FriendInfo Used to store info about a friend's online state and in which room he/she is	49
GameObjectExtensions Small number of extension methods that make it easier for PUN to work cross-Unity-versions. .	50
ExitGames.Client.Photon.GamePropertyKey Class for constants	51
ExitGames.Client.GUI.GizmoTypeDrawer	53
HelpURL Empty implementation of the upcoming HelpURL of Unity 5.1	53
HighlightOwnedGameObj	54
InputToEvent Utility component to forward mouse or touch input to clicked gameobjects	54
InRoomChat	55
InRoomRoundTimer Simple script that uses a property to sync a start time for a multiplayer game	56
IPunCallbacks This interface is used as definition of all callback methods of PUN, except OnPhotonSerialize↔ View	58
IPunObservable Defines the OnPhotonSerializeView method to make it easy to implement correctly for observ- able scripts	67

IPunPrefabPool	
Defines all the methods that a Object Pool must implement, so that PUN can use it	68
ManualPhotonViewAllocator	69
Photon.MonoBehaviour	
This class adds the property photonView, while logging a warning when your game still uses the networkView	70
MoveByKeys	
Very basic component to move a GameObject by WASD and Space	70
OnAwakeUsePhotonView	72
OnClickDestroy	
Implements OnClick to destroy the GameObject it's attached to	73
OnClickInstantiate	74
OnClickLoadSomething	
This component makes it easy to switch scenes or open webpages on click	74
OnJoinedInstantiate	75
OnStartDelete	
This component will destroy the GameObject it is attached to (in Start()).	76
ExitGames.Client.Photon.OperationCode	
Class for constants	76
ExitGames.Client.Photon.LoadbalancingPeer.OpJoinRandomRoomParams	79
ExitGames.Client.Photon.ParameterCode	
Class for constants	80
PhotonAnimatorView	
This class helps you to synchronize Mecanim animations Simply add the component to your GameObject and make sure that the PhotonAnimatorView is added to the list of observed components	90
PhotonAnimatorViewEditor	93
PhotonConverter	93
PhotonEditor	94
PhotonGUI	97
PhotonLagSimulationGui	
This MonoBehaviour is a basic GUI for the Photon client's network-simulation feature	98
PhotonMessageInfo	
Container class for info about a particular message, RPC or update	99
PhotonNetwork	
The main class to use the PhotonNetwork plugin	101
PhotonPingManager	137
PhotonPlayer	
Summarizes a "player" within a room, identified (in that room) by actorID	139
PhotonRigidbody2DView	
This class helps you to synchronize the velocities of a 2d physics Rigidbody	143
PhotonRigidbody2DViewEditor	143
PhotonRigidbodyView	
This class helps you to synchronize the velocities of a physics Rigidbody	144
PhotonRigidbodyViewEditor	144
PhotonStatsGui	
Basic GUI to show traffic and health statistics of the connection to Photon , toggled by shift+tab	145
PhotonStream	
This container is used in OnPhotonSerializeView() to either provide incoming data of a Photon<←View or for you to provide it	146
PhotonStreamQueue	
The PhotonStreamQueue helps you poll object states at higher frequencies then what Photon<←Network.sendRate dictates and then sends all those states at once when Serialize() is called	150
PhotonTransformView	
This class helps you to synchronize position, rotation and scale of a GameObject	152
PhotonTransformViewEditor	153
PhotonTransformViewPositionControl	154
PhotonTransformViewPositionModel	155

PhotonTransformViewRotationControl	158
PhotonTransformViewRotationModel	158
PhotonTransformViewScaleControl	159
PhotonTransformViewScaleModel	159
PhotonView	
PUN's NetworkView replacement class for networking	160
PhotonViewHandler	166
PhotonViewInspector	167
PickupItem	
Makes a scene object pickup-able	167
PickupItemSimple	
Makes a scene object pickup-able	171
PickupItemSyncer	
Finds out which PickupItems are not spawned at the moment and send this to new players	172
PingMonoEditor	
Uses C# Socket class from System.Net.Sockets (as Unity usually does)	173
PointedAtGameObjectInfo	173
Photon.PunBehaviour	
This class provides a .photonView and all callbacks/events that PUN can call	173
PunPlayerScores	183
PunRPC	
Replacement for RPC attribute with different name. Used to flag methods as remote-callable.	183
PunSceneSettings	184
PunTeams	
Implements teams in a room/game with help of player properties	185
PunWizardText	187
QuitOnEscapeOrBack	191
RaiseEventOptions	
Aggregates several less-often used options for operation RaiseEvent . See field descriptions for usage details.	191
Region	193
Room	
This class resembles a room that PUN joins (or joined)	194
RoomInfo	
A simplified room with just the info required to list and join, used for the room listing in the lobby	197
RoomOptions	
Wraps up common room properties needed when you create rooms	201
UnityEngine.SceneManagement.SceneManager	
Minimal implementation of the SceneManager for older Unity, up to v5.2.	203
SceneManagerHelper	203
SceneSetting	204
ScoreExtensions	204
ServerSettings	
Collection of connection-relevant settings, used internally by PhotonNetwork.ConnectUsing()	
Settings	204
ServerSettingsInspector	206
ServerTime	208
ShowInfoOfPlayer	
Can be attached to a GameObject to show info about the owner of the PhotonView	208
ShowStatusWhenConnecting	208
SmoothSyncMovement	209
SupportLogger	210
SupportLogging	210
PhotonAnimatorView.SynchronizedLayer	211
PhotonAnimatorView.SynchronizedParameter	212
TeamExtensions	
Extension used for PunTeams and PhotonPlayer class. Wraps access to the player's custom property.	212

[ExitGames.Client.DemoParticle.TimeKeeper](#)

A utility class that turns it's ShouldExecute property to true after a set interval time has passed [213](#)

[TypedLobby](#)

Refers to a specific lobby (and type) on the server [215](#)

[TypedLobbyInfo](#) [216](#)[WebRpcResponse](#)

Reads an operation response of a WebRpc and provides convenient access to most common values [217](#)

Chapter 5

File Index

5.1 File List

Here is a list of all files with brief descriptions:

C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Editor/Photon↔ Network/ AccountService.cs	219
C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Editor/Photon↔ Network/ PhotonConverter.cs	219
C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Editor/Photon↔ Network/ PhotonEditor.cs	219
C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Editor/Photon↔ Network/ PhotonGUI.cs	219
C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Editor/Photon↔ Network/ PhotonViewHandler.cs	220
C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Editor/Photon↔ Network/ PhotonViewInspector.cs	220
C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Editor/Photon↔ Network/ PhotonViewPrefabApply.cs	220
C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Editor/Photon↔ Network/ PunSceneSettings.cs	220
C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Editor/Photon↔ Network/ ReorderableListResources.cs	220
C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Editor/Photon↔ Network/ ServerSettingsInspector.cs	221
C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Editor/Photon↔ Network/Views/ PhotonAnimatorViewEditor.cs	221
C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Editor/Photon↔ Network/Views/ PhotonRigidbody2DViewEditor.cs	221
C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Editor/Photon↔ Network/Views/ PhotonRigidbodyViewEditor.cs	221
C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Editor/Photon↔ Network/Views/ PhotonTransformViewEditor.cs	221
C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/↔ PhotonNetwork/ CustomTypes.cs	222
C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/↔ PhotonNetwork/ Enums.cs	222
C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/↔ PhotonNetwork/ Extensions.cs	225

C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon PhotonNetwork/ FriendInfo.cs	Unity	Networking/Plugins/↔	225
C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon PhotonNetwork/ GizmoType.cs	Unity	Networking/Plugins/↔	225
C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon PhotonNetwork/ LoadbalancingPeer.cs	Unity	Networking/Plugins/↔	226
C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon PhotonNetwork/ NetworkingPeer.cs	Unity	Networking/Plugins/↔	228
C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon PhotonNetwork/ PhotonClasses.cs Wraps up smaller classes that don't need their own file	Unity	Networking/Plugins/↔	228
C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon PhotonNetwork/ PhotonHandler.cs	Unity	Networking/Plugins/↔	230
C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon PhotonNetwork/ PhotonLagSimulationGui.cs Part of the Optional GUI	Unity	Networking/Plugins/↔	230
C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon PhotonNetwork/ PhotonNetwork.cs	Unity	Networking/Plugins/↔	230
C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon PhotonNetwork/ PhotonPlayer.cs	Unity	Networking/Plugins/↔	231
C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon PhotonNetwork/ PhotonStatsGui.cs Part of the Optional GUI	Unity	Networking/Plugins/↔	231
C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon PhotonNetwork/ PhotonStreamQueue.cs	Unity	Networking/Plugins/↔	232
C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon PhotonNetwork/ PhotonView.cs	Unity	Networking/Plugins/↔	232
C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon PhotonNetwork/ PingCloudRegions.cs	Unity	Networking/Plugins/↔	233
C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon PhotonNetwork/ Room.cs	Unity	Networking/Plugins/↔	233
C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon PhotonNetwork/ RoomInfo.cs	Unity	Networking/Plugins/↔	234
C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon PhotonNetwork/ RPC.cs Reimplements a RPC Attribute, as it's no longer in all versions of the UnityEngine assembly	Unity	Networking/Plugins/↔	234
C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon PhotonNetwork/ RpcIndexComponent.cs Outdated	Unity	Networking/Plugins/↔	234
C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon PhotonNetwork/ ServerSettings.cs ScriptableObject defining a server setup	Unity	Networking/Plugins/↔	234
C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon PhotonNetwork/ SocketUdp.cs	Unity	Networking/Plugins/↔	235
C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon PhotonNetwork/ SocketWebTcp.cs	Unity	Networking/Plugins/↔	235
C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon PhotonNetwork/Views/ PhotonAnimatorView.cs	Unity	Networking/Plugins/↔	235
C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon PhotonNetwork/Views/ PhotonRigidbody2DView.cs	Unity	Networking/Plugins/↔	235
C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon PhotonNetwork/Views/ PhotonRigidbodyView.cs	Unity	Networking/Plugins/↔	236
C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon PhotonNetwork/Views/ PhotonTransformView.cs	Unity	Networking/Plugins/↔	236
C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon PhotonNetwork/Views/ PhotonTransformViewPositionControl.cs	Unity	Networking/Plugins/↔	236
C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon PhotonNetwork/Views/ PhotonTransformViewPositionModel.cs	Unity	Networking/Plugins/↔	236

C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon PhotonNetwork/Views/ PhotonTransformViewRotationControl.cs	Unity	Networking/Plugins/↔	236
C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon PhotonNetwork/Views/ PhotonTransformViewRotationModel.cs	Unity	Networking/Plugins/↔	237
C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon PhotonNetwork/Views/ PhotonTransformViewScaleControl.cs	Unity	Networking/Plugins/↔	237
C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon PhotonNetwork/Views/ PhotonTransformViewScaleModel.cs	Unity	Networking/Plugins/↔	237
C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Scripts/ ConnectAndJoinRandom.cs	Unity	Networking/Utility↔	237
C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Scripts/ HighlightOwnedGameObj.cs	Unity	Networking/Utility↔	237
C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Scripts/ InputToEvent.cs	Unity	Networking/Utility↔	238
C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Scripts/ InRoomChat.cs	Unity	Networking/Utility↔	238
C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Scripts/ InRoomRoundTimer.cs	Unity	Networking/Utility↔	238
C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Scripts/ ManualPhotonViewAllocator.cs	Unity	Networking/Utility↔	238
C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Scripts/ MoveByKeys.cs	Unity	Networking/Utility↔	238
C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Scripts/ OnAwakeUsePhotonView.cs	Unity	Networking/Utility↔	239
C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Scripts/ OnClickDestroy.cs	Unity	Networking/Utility↔	239
C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Scripts/ OnClickInstantiate.cs	Unity	Networking/Utility↔	239
C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Scripts/ OnClickLoadSomething.cs	Unity	Networking/Utility↔	239
C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Scripts/ OnJoinedInstantiate.cs	Unity	Networking/Utility↔	239
C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Scripts/ OnStartDelete.cs	Unity	Networking/Utility↔	239
C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Scripts/ PickupItem.cs	Unity	Networking/Utility↔	240
C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Scripts/ PickupItemSimple.cs	Unity	Networking/Utility↔	240
C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Scripts/ PickupItemSyncer.cs	Unity	Networking/Utility↔	240
C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Scripts/ PointedAtGameObjectInfo.cs	Unity	Networking/Utility↔	240
C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Scripts/ PunPlayerScores.cs	Unity	Networking/Utility↔	241
C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Scripts/ PunTeams.cs	Unity	Networking/Utility↔	241
C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Scripts/ QuitOnEscapeOrBack.cs	Unity	Networking/Utility↔	241
C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Scripts/ ServerTime.cs	Unity	Networking/Utility↔	242
C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Scripts/ ShowInfoOfPlayer.cs	Unity	Networking/Utility↔	242
C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Scripts/ ShowStatusWhenConnecting.cs	Unity	Networking/Utility↔	242
C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Scripts/ SmoothSyncMovement.cs	Unity	Networking/Utility↔	242
C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Scripts/ SupportLogger.cs	Unity	Networking/Utility↔	242

C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Utility↔
Scripts/[TimeKeeper.cs](#) 242

Chapter 6

Module Documentation

6.1 Public API

Groups the most important classes that you need to understand early on.

Classes

- interface [IPunObservable](#)
Defines the `OnPhotonSerializeView` method to make it easy to implement correctly for observable scripts.
- interface [IPunCallbacks](#)
This interface is used as definition of all callback methods of PUN, except `OnPhotonSerializeView`.
- class [Photon.PunBehaviour](#)
This class provides a `.photonView` and all callbacks/events that PUN can call.
- class [PhotonMessageInfo](#)
Container class for info about a particular message, RPC or update.
- class [PhotonStream](#)
This container is used in `OnPhotonSerializeView()` to either provide incoming data of a [PhotonView](#) or for you to provide it.
- class [PhotonNetwork](#)
The main class to use the [PhotonNetwork](#) plugin.
- class [PhotonPlayer](#)
Summarizes a "player" within a room, identified (in that room) by `actorID`.
- class [PhotonView](#)
PUN's `NetworkView` replacement class for networking.
- class [Room](#)
This class resembles a room that PUN joins (or joined).
- class [RoomInfo](#)
A simplified room with just the info required to list and join, used for the room listing in the lobby.

Enumerations

- enum `PhotonNetworkingMessage` {
`PhotonNetworkingMessage.OnConnectedToPhoton`, `PhotonNetworkingMessage.OnLeftRoom`, `PhotonNetworkingMessage.OnMasterClientSwitched`, `PhotonNetworkingMessage.OnPhotonCreateRoomFailed`,
`PhotonNetworkingMessage.OnPhotonJoinRoomFailed`, `PhotonNetworkingMessage.OnCreatedRoom`,
`PhotonNetworkingMessage.OnJoinedLobby`, `PhotonNetworkingMessage.OnLeftLobby`,
`PhotonNetworkingMessage.OnDisconnectedFromPhoton`, `PhotonNetworkingMessage.OnConnectionFail`,
`PhotonNetworkingMessage.OnFailedToConnectToPhoton`, `PhotonNetworkingMessage.OnReceivedRoomListUpdate`,
`PhotonNetworkingMessage.OnJoinedRoom`, `PhotonNetworkingMessage.OnPhotonPlayerConnected`,
`PhotonNetworkingMessage.OnPhotonPlayerDisconnected`, `PhotonNetworkingMessage.OnPhotonRandomJoinFailed`,
`PhotonNetworkingMessage.OnConnectedToMaster`, `PhotonNetworkingMessage.OnPhotonSerializeView`,
`PhotonNetworkingMessage.OnPhotonInstantiate`, `PhotonNetworkingMessage.OnPhotonMaxCcuReached`,
`PhotonNetworkingMessage.OnPhotonCustomRoomPropertiesChanged`, `PhotonNetworkingMessage.OnPhotonPlayerPropertiesChanged`, `PhotonNetworkingMessage.OnUpdatedFriendList`, `PhotonNetworkingMessage.OnCustomAuthenticationFailed`,
`PhotonNetworkingMessage.OnWebRpcResponse`, `PhotonNetworkingMessage.OnOwnershipRequest`,
`PhotonNetworkingMessage.OnLobbyStatisticsUpdate` }
This enum defines the set of MonoMessages Photon Unity Networking is using as callbacks.
- enum `PhotonLogLevel` { `PhotonLogLevel.ErrorsOnly`, `PhotonLogLevel.Informational`, `PhotonLogLevel.Full` }
Used to define the level of logging output created by the PUN classes.
- enum `PhotonTargets` {
`PhotonTargets.All`, `PhotonTargets.Others`, `PhotonTargets.MasterClient`, `PhotonTargets.AllBuffered`,
`PhotonTargets.OthersBuffered`, `PhotonTargets.AllViaServer`, `PhotonTargets.AllBufferedViaServer` }
Enum of "target" options for RPCs.
- enum `PeerState` {
`PeerState.Uninitialized`, `PeerState.PeerCreated`, `PeerState.Queued`, `PeerState.Authenticated`,
`PeerState.JoinedLobby`, `PeerState.DisconnectingFromMasterserver`, `PeerState.ConnectingToGameserver`,
`PeerState.ConnectedToGameserver`,
`PeerState.Joining`, `PeerState.Joined`, `PeerState.Leaving`, `PeerState.DisconnectingFromGameserver`,
`PeerState.ConnectingToMasterserver`, `PeerState.QueuedComingFromGameserver`, `PeerState.Disconnecting`,
`PeerState.Disconnected`,
`PeerState.ConnectedToMaster`, `PeerState.ConnectingToNameServer`, `PeerState.ConnectedToNameServer`,
`PeerState.DisconnectingFromNameServer`,
`PeerState.Authenticating` }
Detailed connection / networking peer state.
- enum `DisconnectCause` {
`DisconnectCause.ExceptionOnConnect` = `StatusCode.ExceptionOnConnect`, `DisconnectCause.SecurityExceptionOnConnect` = `StatusCode.SecurityExceptionOnConnect`, `DisconnectCause.TimeoutDisconnect` = `StatusCode.TimeoutDisconnect`, `DisconnectCause.DisconnectByClientTimeout` = `StatusCode.TimeoutDisconnect`,
`DisconnectCause.InternalReceiveException` = `StatusCode.ExceptionOnReceive`, `DisconnectCause.DisconnectByServer` = `StatusCode.DisconnectByServer`, `DisconnectCause.DisconnectByServerTimeout` = `StatusCode.DisconnectByServer`, `DisconnectCause.DisconnectByServerLogic` = `StatusCode.DisconnectByServerLogic`,
`DisconnectCause.DisconnectByServerUserLimit` = `StatusCode.DisconnectByServerUserLimit`, `DisconnectCause.Exception` = `StatusCode.Exception`, `DisconnectCause.InvalidRegion` = `ErrorCode.InvalidRegion`,
`DisconnectCause.MaxCcuReached` = `ErrorCode.MaxCcuReached`,
`DisconnectCause.InvalidAuthentication` = `ErrorCode.InvalidAuthentication`, `DisconnectCause.AuthenticationTicketExpired` = 32753 }
Summarizes the cause for a disconnect.

Functions

- void `IPunObservable.OnPhotonSerializeView` (`PhotonStream` stream, `PhotonMessageInfo` info)

Called by PUN several times per second, so that your script can write and read synchronization data for the [PhotonView](#).

6.1.1 Detailed Description

Groups the most important classes that you need to understand early on.

6.1.2 Enumeration Type Documentation

6.1.2.1 enum DisconnectCause [strong]

Summarizes the cause for a disconnect.

Used in: OnConnectionFail and OnFailedToConnectToPhoton.

Extracted from the status codes from ExitGames.Client.Photon.StatusCode.

See also

[PhotonNetworkingMessage](#)

Enumerator

ExceptionOnConnect Connection could not be established. Possible cause: Local server not running.

SecurityExceptionOnConnect The security settings for client or server don't allow a connection (see remarks). A common cause for this is that browser clients read a "crossdomain" file from the server. If that file is unavailable or not configured to let the client connect, this exception is thrown. [Photon](#) usually provides this crossdomain file for Unity. If it fails, read: <http://doc.exitgames.com/photon-server/PolicyApp>

TimeoutDisconnect Connection timed out. Possible cause: Remote server not running or required ports blocked (due to router or firewall).

DisconnectByClientTimeout Timeout disconnect by client (which decided an ACK was missing for too long).

InternalReceiveException Exception in the receive-loop. Possible cause: Socket failure.

DisconnectByServer Server actively disconnected this client.

DisconnectByServerTimeout Timeout disconnect by server (which decided an ACK was missing for too long).

DisconnectByServerLogic Server actively disconnected this client. Possible cause: Server's send buffer full (too much data for client).

DisconnectByServerUserLimit Server actively disconnected this client. Possible cause: The server's user limit was hit and client was forced to disconnect (on connect).

Exception Some exception caused the connection to close.

InvalidRegion (32756) Authorization on the [Photon](#) Cloud failed because the app's subscription does not allow to use a particular region's server.

MaxCcuReached (32757) Authorization on the [Photon](#) Cloud failed because the concurrent users (CCU) limit of the app's subscription is reached.

InvalidAuthentication (32767) The [Photon](#) Cloud rejected the sent AppId. Check your Dashboard and make sure the AppId you use is complete and correct.

AuthenticationTicketExpired (32753) The Authentication ticket expired. Handle this by connecting again (which includes an authenticate to get a fresh ticket).

6.1.2.2 enum PeerState [strong]

Detailed connection / networking peer state.

PUN implements a loadbalancing and authentication workflow "behind the scenes", so some states will automatically advance to some follow up state. Those states are commented with "(will-change)".

Enumerator

- Uninitialized** Not running. Only set before initialization and first use.
- PeerCreated** Created and available to connect.
- Queued** Not used at the moment.
- Authenticated** The application is authenticated. PUN usually joins the lobby now.
(will-change) Unless AutoJoinLobby is false.
- JoinedLobby** Client is in the lobby of the Master Server and gets room listings. Use Join, Create or Join↔ Random to get into a room to play.
- DisconnectingFromMasterserver** Disconnecting. (will-change)
- ConnectingToGameserver** Connecting to game server (to join/create a room and play). (will-change)
- ConnectedToGameserver** Similar to Connected state but on game server. Still in process to join/create room.
(will-change)
- Joining** In process to join/create room (on game server). (will-change)
- Joined** Final state of a room join/create sequence. This client can now exchange events / call RPCs with other clients.
- Leaving** Leaving a room. (will-change)
- DisconnectingFromGameserver** Workflow is leaving the game server and will re-connect to the master server. (will-change)
- ConnectingToMasterserver** Workflow is connected to master server and will establish encryption and authenticate your app. (will-change)
- QueuedComingFromGameserver** Same Queued but coming from game server. (will-change)
- Disconnecting** PUN is disconnecting. This leads to Disconnected.
(will-change)
- Disconnected** No connection is setup, ready to connect. Similar to PeerCreated.
- ConnectedToMaster** Final state for connecting to master without joining the lobby (AutoJoinLobby is false).
- ConnectingToNameServer** Client connects to the NameServer. This process includes low level connecting and setting up encryption. When done, state becomes ConnectedToNameServer.
- ConnectedToNameServer** Client is connected to the NameServer and established encryption already. You should call OpGetRegions or ConnectToRegionMaster.
- DisconnectingFromNameServer** When disconnecting from a [Photon](#) NameServer. (will-change)
- Authenticating** When connecting to a [Photon](#) Server, this state is intermediate before you can call any operations. (will-change)

6.1.2.3 enum PhotonLogLevel [strong]

Used to define the level of logging output created by the PUN classes.

Either log errors, info (some more) or full.

Enumerator

- ErrorsOnly** Show only errors. Minimal output. Note: Some might be "runtime errors" which you have to expect.
- Informational** Logs some of the workflow, calls and results.
- Full** Every available log call gets into the console/log. Only use for debugging.

6.1.2.4 enum PhotonNetworkingMessage [strong]

This enum defines the set of MonoMessages [Photon](#) Unity Networking is using as callbacks.

Implemented by PunBehaviour.

Much like "Update()" in Unity, PUN will call methods in specific situations. Often, these methods are triggered when network operations complete (example: when joining a room).

All those methods are defined and described in this enum and implemented by PunBehaviour (which makes it easy to implement them as override).

Each entry is the name of such a method and the description tells you when it gets used by PUN.

Make sure to read the remarks per entry as some methods have optional parameters.

Enumerator

OnConnectedToPhoton Called when the initial connection got established but before you can use the server. OnJoinedLobby() or OnConnectedToMaster() are called when PUN is ready.

This callback is only useful to detect if the server can be reached at all (technically). Most often, it's enough to implement OnFailedToConnectToPhoton() and OnDisconnectedFromPhoton().

OnJoinedLobby() or OnConnectedToMaster() are called when PUN is ready.

When this is called, the low level connection is established and PUN will send your AppId, the user, etc in the background. This is not called for transitions from the masterserver to game servers.

Example: void OnConnectedToPhoton() { ... }

OnLeftRoom Called when the local user/client left a room. When leaving a room, PUN brings you back to the Master Server. Before you can use lobbies and join or create rooms, OnJoinedLobby() or OnConnectedToMaster() will get called again.

Example: void OnLeftRoom() { ... }

OnMasterClientSwitched Called after switching to a new MasterClient when the current one leaves. This is not called when this client enters a room. The former MasterClient is still in the player list when this method get called.

Example: void OnMasterClientSwitched(PhotonPlayer newMasterClient) { ... }

OnPhotonCreateRoomFailed Called when a CreateRoom() call failed. Optional parameters provide ErrorCode and message.

Most likely because the room name is already in use (some other client was faster than you). PUN logs some info if the [PhotonNetwork.logLevel](#) is \geq [PhotonLogLevel.Informational](#).

Example: void OnPhotonCreateRoomFailed() { ... }

Example: void OnPhotonCreateRoomFailed(object[] codeAndMsg) { // codeAndMsg[0] is short ErrorCode. codeAndMsg[1] is string debug msg. }

OnPhotonJoinRoomFailed Called when a JoinRoom() call failed. Optional parameters provide ErrorCode and message.

Most likely error is that the room does not exist or the room is full (some other client was faster than you). PUN logs some info if the [PhotonNetwork.logLevel](#) is \geq [PhotonLogLevel.Informational](#).

Example: void OnPhotonJoinRoomFailed() { ... }

Example: void OnPhotonJoinRoomFailed(object[] codeAndMsg) { // codeAndMsg[0] is short ErrorCode. codeAndMsg[1] is string debug msg. }

OnCreatedRoom Called when this client created a room and entered it. OnJoinedRoom() will be called as well.

This callback is only called on the client which created a room (see [PhotonNetwork.CreateRoom](#)).

As any client might close (or drop connection) anytime, there is a chance that the creator of a room does not execute OnCreatedRoom.

If you need specific room properties or a "start signal", it is safer to implement OnMasterClientSwitched() and to make the new MasterClient check the room's state.

Example: void OnCreatedRoom() { ... }

OnJoinedLobby Called on entering a lobby on the Master Server. The actual room-list updates will call `OnReceivedRoomListUpdate()`.

Note: When `PhotonNetwork.autoJoinLobby` is false, `OnConnectedToMaster()` will be called and the room list won't become available.

While in the lobby, the roomlist is automatically updated in fixed intervals (which you can't modify). The room list gets available when `OnReceivedRoomListUpdate()` gets called after `OnJoinedLobby()`.

Example: `void OnJoinedLobby() { ... }`

OnLeftLobby Called after leaving a lobby. When you leave a lobby, `CreateRoom` and `JoinRandomRoom` automatically refer to the default lobby.

Example: `void OnLeftLobby() { ... }`

OnDisconnectedFromPhoton Called after disconnecting from the `Photon` server. In some cases, other callbacks are called before `OnDisconnectedFromPhoton` is called. Examples: `OnConnectionFail()` and `OnFailedToConnectToPhoton()`.

Example: `void OnDisconnectedFromPhoton() { ... }`

OnConnectionFail Called when something causes the connection to fail (after it was established), followed by a call to `OnDisconnectedFromPhoton()`. If the server could not be reached in the first place, `OnFailedToConnectToPhoton` is called instead. The reason for the error is provided as `StatusCode`.

Example: `void OnConnectionFail(DisconnectCause cause) { ... }`

OnFailedToConnectToPhoton Called if a connect call to the `Photon` server failed before the connection was established, followed by a call to `OnDisconnectedFromPhoton()`. `OnConnectionFail` only gets called when a connection to a `Photon` server was established in the first place.

Example: `void OnFailedToConnectToPhoton(DisconnectCause cause) { ... }`

OnReceivedRoomListUpdate Called for any update of the room-listing while in a lobby (`PhotonNetwork.isInsideLobby`) on the Master Server. PUN provides the list of rooms by `PhotonNetwork.GetRoomList()`. Each item is a `RoomInfo` which might include custom properties (provided you defined those as lobby-listed when creating a room).

Not all types of lobbies provide a listing of rooms to the client. Some are silent and specialized for server-side matchmaking.

Example: `void OnReceivedRoomListUpdate() { ... }`

OnJoinedRoom Called when entering a room (by creating or joining it). Called on all clients (including the Master Client).

This method is commonly used to instantiate player characters. If a match has to be started "actively", you can instead call an `PunRPC` triggered by a user's button-press or a timer.

When this is called, you can usually already access the existing players in the room via `PhotonNetwork.playerList`. Also, all custom properties should be already available as `Room.customProperties`. Check `Room.playerCount` to find out if enough players are in the room to start playing.

Example: `void OnJoinedRoom() { ... }`

OnPhotonPlayerConnected Called when a remote player entered the room. This `PhotonPlayer` is already added to the playerlist at this time.

If your game starts with a certain number of players, this callback can be useful to check the `Room.playerCount` and find out if you can start.

Example: `void OnPhotonPlayerConnected(PhotonPlayer newPlayer) { ... }`

OnPhotonPlayerDisconnected Called when a remote player left the room. This `PhotonPlayer` is already removed from the playerlist at this time.

When your client calls `PhotonNetwork.leaveRoom`, PUN will call this method on the remaining clients. When a remote client drops connection or gets closed, this callback gets executed. after a timeout of several seconds.

Example: `void OnPhotonPlayerDisconnected(PhotonPlayer otherPlayer) { ... }`

OnPhotonRandomJoinFailed Called after a `JoinRandom()` call failed. Optional parameters provide `ErrorCode` and message.

Most likely all rooms are full or no rooms are available. When using multiple lobbies (via `JoinLobby` or `TypedLobby`), another lobby might have more/fitting rooms. PUN logs some info if the `PhotonNetwork.logLevel` is `>= PhotonLogLevel.Informational`.

Example: `void OnPhotonRandomJoinFailed() { ... }`

Example: `void OnPhotonRandomJoinFailed(object[] codeAndMsg) { // codeAndMsg[0] is short Error↵ Code. codeAndMsg[1] is string debug msg. }`

OnConnectedToMaster Called after the connection to the master is established and authenticated but only when [PhotonNetwork.autoJoinLobby](#) is false. If you set [PhotonNetwork.autoJoinLobby](#) to true, `On↵ JoinedLobby()` will be called instead of this.

You can join rooms and create them even without being in a lobby. The default lobby is used in that case. The list of available rooms won't become available unless you join a lobby via `PhotonNetwork.joinLobby`.

Example: `void OnConnectedToMaster() { ... }`

OnPhotonSerializeView Implement to customize the data a [PhotonView](#) regularly synchronizes. Called every 'network-update' when observed by [PhotonView](#).

This method will be called in scripts that are assigned as Observed component of a [PhotonView](#). [PhotonNetwork.sendRateOnSerialize](#) affects how often this method is called. [PhotonNetwork.send↵ Rate](#) affects how often packages are sent by this client.

Implementing this method, you can customize which data a [PhotonView](#) regularly synchronizes. Your code defines what is being sent (content) and how your data is used by receiving clients.

Unlike other callbacks, *OnPhotonSerializeView only gets called when it is assigned to a [PhotonView](#) as [PhotonView.observed](#) script.*

To make use of this method, the [PhotonStream](#) is essential. It will be in "writing" mode" on the client that controls a [PhotonView](#) (`PhotonStream.isWriting == true`) and in "reading mode" on the remote clients that just receive that the controlling client sends.

If you skip writing any value into the stream, PUN will skip the update. Used carefully, this can conserve bandwidth and messages (which have a limit per room/second).

Note that `OnPhotonSerializeView` is not called on remote clients when the sender does not send any update. This can't be used as "x-times per second Update()".

Example: `void OnPhotonSerializeView(PhtonStream stream, PhotonMessageInfo info) { ... }`

OnPhotonInstantiate Called on all scripts on a [GameObject](#) (and children) that have been Instantiated using [PhotonNetwork.Instantiate](#). [PhotonMessageInfo](#) parameter provides info about who created the object and when (based off `PhotonNetworking.time`).

Example: `void OnPhotonInstantiate(PhtonMessageInfo info) { ... }`

OnPhotonMaxCccuReached Because the concurrent user limit was (temporarily) reached, this client is rejected by the server and disconnecting. When this happens, the user might try again later. You can't create or join rooms in `OnPhotonMaxCcuReached()`, cause the client will be disconnecting. You can raise the CCU limits with a new license (when you host yourself) or extended subscription (when using the [Photon](#) Cloud). The [Photon](#) Cloud will mail you when the CCU limit was reached. This is also visible in the Dashboard (webpage).

Example: `void OnPhotonMaxCccuReached() { ... }`

OnPhotonCustomRoomPropertiesChanged Called when a room's custom properties changed. The `propertiesThatChanged` contains all that was set via [Room.SetCustomProperties](#).

Since v1.25 this method has one parameter: `Hashtable propertiesThatChanged`. Changing properties must be done by [Room.SetCustomProperties](#), which causes this callback locally, too.

Example: `void OnPhotonCustomRoomPropertiesChanged(Hashtable propertiesThatChanged) { ... }`

OnPhotonPlayerPropertiesChanged Called when custom player-properties are changed. Player and the changed properties are passed as `object[]`.

Since v1.25 this method has one parameter: `object[] playerAndUpdatedProps`, which contains two entries.

[0] is the affected [PhotonPlayer](#).

[1] is the `Hashtable` of properties that changed.

We are using a `object[]` due to limitations of Unity's `GameObject.SendMessage` (which has only one optional parameter).

Changing properties must be done by [PhotonPlayer.SetCustomProperties](#), which causes this callback locally, too.

Example:

```
void OnPhotonPlayerPropertiesChanged(object[] playerAndUpdatedProps) {
    PhotonPlayer player = playerAndUpdatedProps[0] as PhotonPlayer;
    Hashtable props = playerAndUpdatedProps[1] as Hashtable;
    //...
}
```

OnUpdatedFriendList Called when the server sent the response to a FindFriends request and updated [PhotonNetwork.Friends](#). The friends list is available as [PhotonNetwork.Friends](#), listing name, online state and the room a user is in (if any).

Example: void OnUpdatedFriendList() { ... }

OnCustomAuthenticationFailed Called when the custom authentication failed. Followed by disconnect!

Custom Authentication can fail due to user-input, bad tokens/secrets. If authentication is successful, this method is not called. Implement OnJoinedLobby() or OnConnectedToMaster() (as usual).

During development of a game, it might also fail due to wrong configuration on the server side. In those cases, logging the debugMessage is very important.

Unless you setup a custom authentication service for your app (in the [Dashboard](#)), this won't be called!

Example: void OnCustomAuthenticationFailed(string debugMessage) { ... }

OnWebRpcResponse Called by PUN when the response to a WebRPC is available. See [PhotonNetwork.WebRPC](#).

Important: The response.ReturnCode is 0 if [Photon](#) was able to reach your web-service. The content of the response is what your web-service sent. You can create a WebResponse instance from it. Example: [WebRpcResponse](#) webResponse = new [WebRpcResponse\(operationResponse\)](#);

Please note: Class OperationResponse is in a namespace which needs to be "used": using [ExitGames.Client.Photon](#); // includes OperationResponse (and other classes)

The OperationResponse.ReturnCode by [Photon](#) is: 0 for "OK" -3 for "Web-Service not configured" (see [Dashboard / WebHooks](#)) -5 for "Web-Service does now have RPC path/name" (at least for Azure)

Example: void OnWebRpcResponse(OperationResponse response) { ... }

OnOwnershipRequest Called when another player requests ownership of a [PhotonView](#) from you (the current owner). The parameter viewAndPlayer contains:

[PhotonView](#) view = viewAndPlayer[0] as [PhotonView](#);

[PhotonPlayer](#) requestingPlayer = viewAndPlayer[1] as [PhotonPlayer](#);

void OnOwnershipRequest(object[] viewAndPlayer) {} //

OnLobbyStatisticsUpdate Called when the Master Server sent an update for the Lobby Statistics, updating [PhotonNetwork.LobbyStatistics](#). This callback has two preconditions: EnableLobbyStatistics must be set to true, before this client connects. And the client has to be connected to the Master Server, which is providing the info about lobbies.

6.1.2.5 enum PhotonTargets [strong]

Enum of "target" options for RPCs.

These define which remote clients get your RPC call.

Enumerator

All Sends the RPC to everyone else and executes it immediately on this client. Player who join later will not execute this RPC.

Others Sends the RPC to everyone else. This client does not execute the RPC. Player who join later will not execute this RPC.

MasterClient Sends the RPC to MasterClient only. Careful: The MasterClient might disconnect before it executes the RPC and that might cause dropped RPCs.

AllBuffered Sends the RPC to everyone else and executes it immediately on this client. New players get the RPC when they join as it's buffered (until this client leaves).

OthersBuffered Sends the RPC to everyone. This client does not execute the RPC. New players get the RPC when they join as it's buffered (until this client leaves).

AllViaServer Sends the RPC to everyone (including this client) through the server. This client executes the RPC like any other when it received it from the server. Benefit: The server's order of sending the RPCs is the same on all clients.

AllBufferedViaServer Sends the RPC to everyone (including this client) through the server and buffers it for players joining later. This client executes the RPC like any other when it received it from the server. Benefit: The server's order of sending the RPCs is the same on all clients.

6.1.3 Function Documentation

6.1.3.1 void IPunObservable.OnPhotonSerializeView (PhotonStream stream, PhotonMessageInfo info)

Called by PUN several times per second, so that your script can write and read synchronization data for the [PhotonView](#).

This method will be called in scripts that are assigned as Observed component of a [PhotonView](#).

[PhotonNetwork.sendRateOnSerialize](#) affects how often this method is called.

[PhotonNetwork.sendRate](#) affects how often packages are sent by this client.

Implementing this method, you can customize which data a [PhotonView](#) regularly synchronizes. Your code defines what is being sent (content) and how your data is used by receiving clients.

Unlike other callbacks, *OnPhotonSerializeView* only gets called when it is assigned to a [PhotonView](#) as [PhotonView.observed](#) script.

To make use of this method, the [PhotonStream](#) is essential. It will be in "writing" mode" on the client that controls a PhotonView ([PhotonStream.isWriting](#) == true) and in "reading mode" on the remote clients that just receive that the controlling client sends.

If you skip writing any value into the stream, PUN will skip the update. Used carefully, this can conserve bandwidth and messages (which have a limit per room/second).

Note that *OnPhotonSerializeView* is not called on remote clients when the sender does not send any update. This can't be used as "x-times per second Update()".

Implemented in [PhotonTransformView](#), and [PickupItem](#).

6.2 Optional Gui Elements

Useful GUI elements for PUN.

Classes

- class [PhotonLagSimulationGui](#)
This MonoBehaviour is a basic GUI for the [Photon](#) client's network-simulation feature.
- class [PhotonStatsGui](#)
Basic GUI to show traffic and health statistics of the connection to [Photon](#), toggled by shift+tab.

6.2.1 Detailed Description

Useful GUI elements for PUN.

Chapter 7

Namespace Documentation

7.1 ExitGames Namespace Reference

Namespaces

- namespace [Client](#)

7.2 ExitGames.Client Namespace Reference

Namespaces

- namespace [DemoParticle](#)
- namespace [GUI](#)
- namespace [Photon](#)

7.3 ExitGames.Client.DemoParticle Namespace Reference

Classes

- class [TimeKeeper](#)

A utility class that turns it's ShouldExecute property to true after a set interval time has passed.

7.4 ExitGames.Client.GUI Namespace Reference

Classes

- class [GizmoTypeDrawer](#)

Enumerations

- enum [GizmoType](#) { [GizmoType.WireSphere](#), [GizmoType.Sphere](#), [GizmoType.WireCube](#), [GizmoType.Cube](#) }

7.4.1 Enumeration Type Documentation

7.4.1.1 enum ExitGames.Client.GUI.GizmoType [strong]

Enumerator

WireSphere
Sphere
WireCube
Cube

7.5 ExitGames.Client.Photon Namespace Reference

Classes

- class [ActorProperties](#)
Class for constants.
- class [ErrorCode](#)
[ErrorCode](#) defines the default codes associated with [Photon](#) client/server communication.
- class [EventCode](#)
Class for constants.
- class [GamePropertyKey](#)
Class for constants.
- class **LoadbalancingPeer**
Internally used by PUN, a [LoadbalancingPeer](#) provides the operations and enum definitions needed to use the [Photon](#) Loadbalancing server (or the [Photon](#) Cloud).
- class [OperationCode](#)
Class for constants.
- class [ParameterCode](#)
Class for constants.
- class **SocketUdp**
Internal class to encapsulate the network i/o functionality for the realtime library.

Enumerations

- enum [JoinMode](#) : byte { [JoinMode.Default](#) = 0, [JoinMode.CreateIfNotExists](#) = 1, [JoinMode.JoinOrRejoin](#) = 2, [JoinMode.RejoinOnly](#) = 3 }
Defines possible values for [OpJoinRoom](#) and [OpJoinOrCreate](#).
- enum [MatchmakingMode](#) : byte { [MatchmakingMode.FillRoom](#) = 0, [MatchmakingMode.SerialMatching](#) = 1, [MatchmakingMode.RandomMatching](#) = 2 }
Options for matchmaking rules for [OpJoinRandom](#).
- enum [ReceiverGroup](#) : byte { [ReceiverGroup.Others](#) = 0, [ReceiverGroup.All](#) = 1, [ReceiverGroup.MasterClient](#) = 2 }
Lite - [OpRaiseEvent](#) lets you chose which actors in the room should receive events.
- enum [EventCaching](#) : byte { [EventCaching.DoNotCache](#) = 0, [EventCaching.MergeCache](#) = 1, [EventCaching.ReplaceCache](#) = 2, [EventCaching.RemoveCache](#) = 3, [EventCaching.AddToRoomCache](#) = 4, [EventCaching.AddToRoomCacheGlobal](#) = 5, [EventCaching.RemoveFromRoomCache](#) = 6, [EventCaching.RemoveFromRoomCacheForActorsLeft](#) = 7, [EventCaching.SliceIncreaseIndex](#) = 10, [EventCaching.SliceSetIndex](#) = 11, [EventCaching.SlicePurgeIndex](#) = 12, [EventCaching.SlicePurgeUpToIndex](#) = 13 }
Lite - [OpRaiseEvent](#) allows you to cache events and automatically send them to joining players in a room.
- enum [PropertyTypeFlag](#) : byte { [PropertyTypeFlag.None](#) = 0x00, [PropertyTypeFlag.Game](#) = 0x01, [PropertyTypeFlag.Actor](#) = 0x02, [PropertyTypeFlag.GameAndActor](#) = Game | Actor }
Flags for "types of properties", being used as filter in [OpGetProperties](#).

7.5.1 Enumeration Type Documentation

7.5.1.1 enum ExitGames.Client.Photon.EventCaching : byte [strong]

Lite - OpRaiseEvent allows you to cache events and automatically send them to joining players in a room.

Events are cached per event code and player: Event 100 (example!) can be stored once per player. Cached events can be modified, replaced and removed.

Caching works only combination with ReceiverGroup options Others and All.

Enumerator

DoNotCache Default value (not sent).

MergeCache Will merge this event's keys with those already cached.

ReplaceCache Replaces the event cache for this eventCode with this event's content.

RemoveCache Removes this event (by eventCode) from the cache.

AddToRoomCache Adds an event to the room's cache

AddToRoomCacheGlobal Adds this event to the cache for actor 0 (becoming a "globally owned" event in the cache).

RemoveFromRoomCache Remove fitting event from the room's cache.

RemoveFromRoomCacheForActorsLeft Removes events of players who already left the room (cleaning up).

SliceIncreaseIndex Increase the index of the sliced cache.

SliceSetIndex Set the index of the sliced cache. You must set RaiseEventOptions.CacheSliceIndex for this.

SlicePurgeIndex Purge cache slice with index. Exactly one slice is removed from cache. You must set RaiseEventOptions.CacheSliceIndex for this.

SlicePurgeUpToIndex Purge cache slices with specified index and anything lower than that. You must set RaiseEventOptions.CacheSliceIndex for this.

7.5.1.2 enum ExitGames.Client.Photon.JoinMode : byte [strong]

Defines possible values for OpJoinRoom and OpJoinOrCreate.

It tells the server if the room can be only be joined normally, created implicitly or found on a web-service for Turn-based games.

These values are not directly used by a game but implicitly set.

Enumerator

Default Regular join. The room must exist.

CreateIfNotExists Join or create the room if it's not existing. Used for OpJoinOrCreate for example.

JoinOrRejoin The room might be out of memory and should be loaded (if possible) from a Turnbased web-service.

RejoinOnly Only re-join will be allowed. If the user is not yet in the room, this will fail.

7.5.1.3 enum `ExitGames.Client.Photon.MatchmakingMode` : `byte` [`strong`]

Options for matchmaking rules for `OpJoinRandom`.

Enumerator

FillRoom Fills up rooms (oldest first) to get players together as fast as possible. Default.

Makes most sense with `MaxPlayers > 0` and games that can only start with more players.

SerialMatching Distributes players across available rooms sequentially but takes filter into account. Without filter, rooms get players evenly distributed.

RandomMatching Joins a (fully) random room. Expected properties must match but aside from this, any available room might be selected.

7.5.1.4 enum `ExitGames.Client.Photon.PropertyTypeFlag` : `byte` [`strong`]

Flags for "types of properties", being used as filter in `OpGetProperties`.

Enumerator

None (0x00) Flag type for no property type.

Game (0x01) Flag type for game-attached properties.

Actor (0x02) Flag type for actor related properties.

GameAndActor (0x01) Flag type for game AND actor properties. Equal to 'Game'

7.5.1.5 enum `ExitGames.Client.Photon.ReceiverGroup` : `byte` [`strong`]

Lite - `OpRaiseEvent` lets you chose which actors in the room should receive events.

By default, events are sent to "Others" but you can overrule this.

Enumerator

Others Default value (not sent). Anyone else gets my event.

All Everyone in the current room (including this peer) will get this event.

MasterClient The server sends this event only to the actor with the lowest `actorNumber`. The "master client" does not have special rights but is the one who is in this room the longest time.

7.6 Photon Namespace Reference

Classes

- class [MonoBehaviour](#)

This class adds the property `photonView`, while logging a warning when your game still uses the `networkView`.

- class [PunBehaviour](#)

This class provides a `.photonView` and all callbacks/events that `PUN` can call.

Typedefs

- using [Hashtable](#) = ExitGames.Client.Photon.Hashtable

7.6.1 Typedef Documentation

7.6.1.1 using Photon.Hashtable = typedef ExitGames.Client.Photon.Hashtable

7.7 Rotorz Namespace Reference

Namespaces

- namespace [ReorderableList](#)

7.8 Rotorz.ReorderableList Namespace Reference

Namespaces

- namespace [Internal](#)

7.9 Rotorz.ReorderableList.Internal Namespace Reference

Classes

- class **ReorderableListResources**
Resources to assist with reorderable list control.

7.10 UnityEngine Namespace Reference

Namespaces

- namespace [SceneManagement](#)

7.11 UnityEngine.SceneManagement Namespace Reference

Classes

- class [SceneManager](#)
Minimal implementation of the [SceneManager](#) for older Unity, up to v5.2.

Chapter 8

Class Documentation

8.1 AccountService Class Reference

Public Types

- enum [Origin](#) : byte { [Origin.ServerWeb](#) = 1, [Origin.CloudWeb](#) = 2, [Origin.Pun](#) = 3, [Origin.Playmaker](#) = 4 }

Public Member Functions

- [AccountService](#) ()
Creates a instance of the Account Service to register [Photon](#) Cloud accounts.
- void [RegisterByEmail](#) (string email, [Origin](#) origin)
Attempts to create a [Photon](#) Cloud Account.
- void [RegisterByEmailAsync](#) (string email, [Origin](#) origin, Action< [AccountService](#) > callback=null)
Attempts to create a [Photon](#) Cloud Account asynchronously.

Static Public Member Functions

- static bool [Validator](#) (object sender, X509Certificate certificate, X509Chain chain, SslPolicyErrors policy↔ Errors)

Properties

- string [Message](#) [get]
- string [Appld](#) [get]
- int [ReturnCode](#) [get]

8.1.1 Member Enumeration Documentation

8.1.1.1 enum AccountService.Origin : byte [strong]

Enumerator

[ServerWeb](#)

[CloudWeb](#)

[Pun](#)

[Playmaker](#)

8.1.2 Constructor & Destructor Documentation

8.1.2.1 `AccountService.AccountService ()`

Creates a instance of the Account Service to register [Photon](#) Cloud accounts.

8.1.3 Member Function Documentation

8.1.3.1 `void AccountService.RegisterByEmail (string email, Origin origin)`

Attempts to create a [Photon](#) Cloud Account.

Check ReturnCode, Message and Appld to get the result of this attempt.

Parameters

<i>email</i>	Email of the account.
<i>origin</i>	Marks which channel created the new account (if it's new).

8.1.3.2 `void AccountService.RegisterByEmailAsync (string email, Origin origin, Action< AccountService > callback = null)`

Attempts to create a [Photon](#) Cloud Account asynchronously.

Once your callback is called, check ReturnCode, Message and Appld to get the result of this attempt.

Parameters

<i>email</i>	Email of the account.
<i>origin</i>	Marks which channel created the new account (if it's new).
<i>callback</i>	Called when the result is available.

8.1.3.3 `static bool AccountService.Validator (object sender, X509Certificate certificate, X509Chain chain, SslPolicyErrors policyErrors) [static]`

8.1.4 Property Documentation

8.1.4.1 `string AccountService.Appld [get]`

8.1.4.2 `string AccountService.Message [get]`

8.1.4.3 `int AccountService.ReturnCode [get]`

The documentation for this class was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Editor/Photon↔ Network/[AccountService.cs](#)

8.2 ExitGames.Client.Photon.ActorProperties Class Reference

Class for constants.

Public Attributes

- const byte `PlayerName` = 255
(255) Name of a player/actor.
- const byte `IsInactive` = 254
(254) Tells you if the player is currently in this game (getting events live).
- const byte `UserId` = 253
UserId of the player. Sent when room gets created with `RoomOptions.publishUserId = true`.

8.2.1 Detailed Description

Class for constants.

These (byte) values define "well known" properties for an Actor / Player. Pun uses these constants internally.

"Custom properties" have to use a string-type as key. They can be assigned at will.

8.2.2 Member Data Documentation

8.2.2.1 const byte ExitGames.Client.Photon.ActorProperties.IsInactive = 254

(254) Tells you if the player is currently in this game (getting events live).

A server-set value for async games, where players can leave the game and return later.

8.2.2.2 const byte ExitGames.Client.Photon.ActorProperties.PlayerName = 255

(255) Name of a player/actor.

8.2.2.3 const byte ExitGames.Client.Photon.ActorProperties.UserId = 253

UserId of the player. Sent when room gets created with `RoomOptions.publishUserId = true`.

The documentation for this class was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/Photon↔ Network/[LoadbalancingPeer.cs](#)

8.3 AuthenticationValues Class Reference

Container for user authentication in [Photon](#).

Public Member Functions

- [AuthenticationValues](#) ()
Creates empty auth values without any info.
- [AuthenticationValues](#) (string userId)
Creates minimal info about the user.
- virtual void [SetAuthPostData](#) (string stringData)
Sets the data to be passed-on to the auth service via POST.
- virtual void [SetAuthPostData](#) (byte[] byteData)
Sets the data to be passed-on to the auth service via POST.
- virtual void [AddAuthParameter](#) (string key, string value)
Adds a key-value pair to the get-parameters used for Custom Auth.
- override string [ToString](#) ()

Properties

- [CustomAuthenticationType AuthType](#) [get, set]
The type of custom authentication provider that should be used. Currently only "Custom" or "None" (turns this off).
- string [AuthGetParameters](#) [get, set]
This string must contain any (http get) parameters expected by the used authentication service.
- object [AuthPostData](#) [get]
Data to be passed-on to the auth service via POST. Default: null (not sent). Either string or byte[] (see setters).
- string [Token](#) [get, set]
After initial authentication, [Photon](#) provides a token for this client / user, which is subsequently used as (cached) validation.
- string [UserId](#) [get, set]
The UserId should be a unique identifier per user. This is for finding friends, etc..

8.3.1 Detailed Description

Container for user authentication in [Photon](#).

Set AuthValues before you connect - all else is handled.

On [Photon](#), user authentication is optional but can be useful in many cases. If you want to FindFriends, a unique ID per user is very practical.

There are basically three options for user authentication: None at all, the client sets some UserId or you can use some account web-service to authenticate a user (and set the UserId server-side).

Custom Authentication lets you verify end-users by some kind of login or token. It sends those values to [Photon](#) which will verify them before granting access or disconnecting the client.

The [Photon](#) Cloud Dashboard will let you enable this feature and set important server values for it. <https://www.exitgames.com/dashboard>

8.3.2 Constructor & Destructor Documentation

8.3.2.1 AuthenticationValues.AuthenticationValues ()

Creates empty auth values without any info.

8.3.2.2 AuthenticationValues.AuthenticationValues (string userId)

Creates minimal info about the user.

If this is authenticated or not, depends on the set AuthType.

Parameters

<i>userId</i>	Some UserId to set in Photon .
---------------	--

8.3.3 Member Function Documentation

8.3.3.1 virtual void AuthenticationValues.AddAuthParameter (string key, string value) [virtual]

Adds a key-value pair to the get-parameters used for Custom Auth.

This method does uri-encoding for you.

Parameters

<i>key</i>	Key for the value to set.
<i>value</i>	Some value relevant for Custom Authentication.

8.3.3.2 virtual void AuthenticationValues.SetAuthPostData (string stringData) [virtual]

Sets the data to be passed-on to the auth service via POST.

Parameters

<i>stringData</i>	String data to be sent in the body of the POST request. An empty string will set the post data to null.
-------------------	---

8.3.3.3 virtual void AuthenticationValues.SetAuthPostData (byte[] byteData) [virtual]

Sets the data to be passed-on to the auth service via POST.

Parameters

<i>byteData</i>	Binary token / auth-data to pass on.
-----------------	--------------------------------------

8.3.3.4 override string AuthenticationValues.ToString ()

8.3.4 Property Documentation

8.3.4.1 string AuthenticationValues.AuthGetParameters [get], [set]

This string must contain any (http get) parameters expected by the used authentication service.

By default, username and token.

Standard http get parameters are used here and passed on to the service that's defined in the server ([Photon](#) Cloud Dashboard).

8.3.4.2 object `AuthenticationValues.AuthPostData` `[get]`

Data to be passed-on to the auth service via POST. Default: null (not sent). Either string or byte[] (see setters).

8.3.4.3 CustomAuthenticationType `AuthenticationValues.AuthType` `[get]`, `[set]`

The type of custom authentication provider that should be used. Currently only "Custom" or "None" (turns this off).

8.3.4.4 string `AuthenticationValues.Token` `[get]`, `[set]`

After initial authentication, [Photon](#) provides a token for this client / user, which is subsequently used as (cached) validation.

8.3.4.5 string `AuthenticationValues.UserId` `[get]`, `[set]`

The UserId should be a unique identifier per user. This is for finding friends, etc..

The documentation for this class was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/Photon↔ Network/[LoadbalancingPeer.cs](#)

8.4 ConnectAndJoinRandom Class Reference

This script automatically connects to [Photon](#) (using the settings file), tries to join a random room and creates one if none was found (which is ok).

Inherits [Photon.MonoBehaviour](#).

Public Member Functions

- virtual void [Start](#) ()
- virtual void [Update](#) ()
- virtual void [OnConnectedToMaster](#) ()
- virtual void [OnJoinedLobby](#) ()
- virtual void [OnPhotonRandomJoinFailed](#) ()
- virtual void [OnFailedToConnectToPhoton](#) ([DisconnectCause](#) cause)
- void [OnJoinedRoom](#) ()

Public Attributes

- bool [AutoConnect](#) = true
Connect automatically? If false you can set this to true later on or call [ConnectUsingSettings](#) in your own scripts.
- byte [Version](#) = 1

Additional Inherited Members

8.4.1 Detailed Description

This script automatically connects to [Photon](#) (using the settings file), tries to join a random room and creates one if none was found (which is ok).

8.4.2 Member Function Documentation

8.4.2.1 `virtual void ConnectAndJoinRandom.OnConnectedToMaster () [virtual]`

8.4.2.2 `virtual void ConnectAndJoinRandom.OnFailedToConnectToPhoton (DisconnectCause cause) [virtual]`

8.4.2.3 `virtual void ConnectAndJoinRandom.OnJoinedLobby () [virtual]`

8.4.2.4 `void ConnectAndJoinRandom.OnJoinedRoom ()`

8.4.2.5 `virtual void ConnectAndJoinRandom.OnPhotonRandomJoinFailed () [virtual]`

8.4.2.6 `virtual void ConnectAndJoinRandom.Start () [virtual]`

8.4.2.7 `virtual void ConnectAndJoinRandom.Update () [virtual]`

8.4.3 Member Data Documentation

8.4.3.1 `bool ConnectAndJoinRandom.AutoConnect = true`

Connect automatically? If false you can set this to true later on or call `ConnectUsingSettings` in your own scripts.

8.4.3.2 `byte ConnectAndJoinRandom.Version = 1`

The documentation for this class was generated from the following file:

- `C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/UtilityScripts/Connect↵AndJoinRandom.cs`

8.5 ExitGames.Client.Photon.LoadbalancingPeer.EnterRoomParams Class Reference

Public Attributes

- `string RoomName`
- `RoomOptions RoomOptions`
- `TypedLobby Lobby`
- `Hashtable PlayerProperties`
- `bool OnGameServer = true`
- `bool CreateIfNotExists`

8.5.1 Member Data Documentation

8.5.1.1 `bool ExitGames.Client.Photon.LoadbalancingPeer.EnterRoomParams.CreateIfNotExists`

8.5.1.2 `TypedLobby ExitGames.Client.Photon.LoadbalancingPeer.EnterRoomParams.Lobby`

8.5.1.3 `bool ExitGames.Client.Photon.LoadbalancingPeer.EnterRoomParams.OnGameServer = true`

8.5.1.4 `Hashtable ExitGames.Client.Photon.LoadbalancingPeer.EnterRoomParams.PlayerProperties`

8.5.1.5 `string ExitGames.Client.Photon.LoadbalancingPeer.EnterRoomParams.RoomName`

8.5.1.6 `RoomOptions ExitGames.Client.Photon.LoadbalancingPeer.EnterRoomParams.RoomOptions`

The documentation for this class was generated from the following file:

- `C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/PhotonNetwork/LoadbalancingPeer.cs`

8.6 ExitGames.Client.Photon.ErrorCode Class Reference

`ErrorCode` defines the default codes associated with `Photon` client/server communication.

Public Attributes

- `const int Ok = 0`
(0) is always "OK", anything else an error or specific situation.
- `const int OperationNotAllowedInCurrentState = -3`
(-3) Operation can't be executed yet (e.g.
- `const int InvalidOperationCode = -2`
(-2) The operation you called is not implemented on the server (application) you connect to. Make sure you run the fitting applications.
- `const int InvalidOperation = -2`
(-2) The operation you called could not be executed on the server.
- `const int InternalServerError = -1`
(-1) Something went wrong in the server. Try to reproduce and contact Exit Games.
- `const int InvalidAuthentication = 0x7FFF`
(32767) Authentication failed. Possible cause: AppId is unknown to `Photon` (in cloud service).
- `const int GamelIdAlreadyExists = 0x7FFF - 1`
(32766) GamelId (name) already in use (can't create another). Change name.
- `const int GameFull = 0x7FFF - 2`
(32765) Game is full. This rarely happens when some player joined the room before your join completed.
- `const int GameClosed = 0x7FFF - 3`
(32764) Game is closed and can't be joined. Join another game.
- `const int AlreadyMatched = 0x7FFF - 4`
- `const int ServerFull = 0x7FFF - 5`
(32762) Not in use currently.

- const int [UserBlocked](#) = 0x7FFF - 6
(32761) Not in use currently.
- const int [NoRandomMatchFound](#) = 0x7FFF - 7
(32760) Random matchmaking only succeeds if a room exists that's neither closed nor full. Repeat in a few seconds or create a new room.
- const int [GameDoesNotExist](#) = 0x7FFF - 9
(32758) Join can fail if the room (name) is not existing (anymore). This can happen when players leave while you join.
- const int [MaxCcuReached](#) = 0x7FFF - 10
(32757) Authorization on the [Photon](#) Cloud failed because the concurrent users (CCU) limit of the app's subscription is reached.
- const int [InvalidRegion](#) = 0x7FFF - 11
(32756) Authorization on the [Photon](#) Cloud failed because the app's subscription does not allow to use a particular region's server.
- const int [CustomAuthenticationFailed](#) = 0x7FFF - 12
(32755) Custom Authentication of the user failed due to setup reasons (see Cloud Dashboard) or the provided user data (like username or token).
- const int [AuthenticationTicketExpired](#) = 0x7FFF - 13
(32753) The Authentication ticket expired. Usually, this is refreshed behind the scenes. Connect (and authorize) again.
- const int [PluginReportedError](#) = 0x7FFF - 15
(32752) A server-side plugin (or webhook) failed to execute and reported an error.
- const int [PluginMismatch](#) = 0x7FFF - 16
(32751) CreateGame/JoinGame/Join operation fails if expected plugin does not correspond to loaded one.

8.6.1 Detailed Description

[ErrorCode](#) defines the default codes associated with [Photon](#) client/server communication.

8.6.2 Member Data Documentation

8.6.2.1 const int ExitGames.Client.Photon.ErrorCode.AlreadyMatched = 0x7FFF - 4

8.6.2.2 const int ExitGames.Client.Photon.ErrorCode.AuthenticationTicketExpired = 0x7FFF - 13

(32753) The Authentication ticket expired. Usually, this is refreshed behind the scenes. Connect (and authorize) again.

8.6.2.3 const int ExitGames.Client.Photon.ErrorCode.CustomAuthenticationFailed = 0x7FFF - 12

(32755) Custom Authentication of the user failed due to setup reasons (see Cloud Dashboard) or the provided user data (like username or token).

Check error message for details.

8.6.2.4 const int ExitGames.Client.Photon.ErrorCode.GameClosed = 0x7FFF - 3

(32764) Game is closed and can't be joined. Join another game.

8.6.2.5 `const int ExitGames.Client.Photon.ErrorCode.GameDoesNotExist = 0x7FFF - 9`

(32758) Join can fail if the room (name) is not existing (anymore). This can happen when players leave while you join.

8.6.2.6 `const int ExitGames.Client.Photon.ErrorCode.GameFull = 0x7FFF - 2`

(32765) Game is full. This rarely happens when some player joined the room before your join completed.

8.6.2.7 `const int ExitGames.Client.Photon.ErrorCode.GameIdAlreadyExists = 0x7FFF - 1`

(32766) GameId (name) already in use (can't create another). Change name.

8.6.2.8 `const int ExitGames.Client.Photon.ErrorCode.InternalServerError = -1`

(-1) Something went wrong in the server. Try to reproduce and contact Exit Games.

8.6.2.9 `const int ExitGames.Client.Photon.ErrorCode.InvalidAuthentication = 0x7FFF`

(32767) Authentication failed. Possible cause: AppId is unknown to [Photon](#) (in cloud service).

8.6.2.10 `const int ExitGames.Client.Photon.ErrorCode.InvalidOperation = -2`

(-2) The operation you called could not be executed on the server.

Make sure you are connected to the server you expect.

This code is used in several cases: The arguments/parameters of the operation might be out of range, missing entirely or conflicting. The operation you called is not implemented on the server (application). Server-side plugins affect the available operations.

8.6.2.11 `const int ExitGames.Client.Photon.ErrorCode.InvalidOperationCode = -2`

(-2) The operation you called is not implemented on the server (application) you connect to. Make sure you run the fitting applications.

8.6.2.12 `const int ExitGames.Client.Photon.ErrorCode.InvalidRegion = 0x7FFF - 11`

(32756) Authorization on the [Photon](#) Cloud failed because the app's subscription does not allow to use a particular region's server.

Some subscription plans for the [Photon](#) Cloud are region-bound. Servers of other regions can't be used then. Check your master server address and compare it with your [Photon](#) Cloud Dashboard's info. <https://cloud.photonengine.com/dashboard>

OpAuthorize is part of connection workflow but only on the [Photon](#) Cloud, this error can happen. Self-hosted [Photon](#) servers with a CCU limited license won't let a client connect at all.

8.6.2.13 `const int ExitGames.Client.Photon.ErrorCode.MaxCcuReached = 0x7FFF - 10`

(32757) Authorization on the [Photon](#) Cloud failed because the concurrent users (CCU) limit of the app's subscription is reached.

Unless you have a plan with "CCU Burst", clients might fail the authentication step during connect. Affected client are unable to call operations. Please note that players who end a game and return to the master server will disconnect and re-connect, which means that they just played and are rejected in the next minute / re-connect. This is a temporary measure. Once the CCU is below the limit, players will be able to connect and play again.

OpAuthorize is part of connection workflow but only on the [Photon](#) Cloud, this error can happen. Self-hosted [Photon](#) servers with a CCU limited license won't let a client connect at all.

8.6.2.14 `const int ExitGames.Client.Photon.ErrorCode.NoRandomMatchFound = 0x7FFF - 7`

(32760) Random matchmaking only succeeds if a room exists that's neither closed nor full. Repeat in a few seconds or create a new room.

8.6.2.15 `const int ExitGames.Client.Photon.ErrorCode.Ok = 0`

(0) is always "OK", anything else is an error or specific situation.

8.6.2.16 `const int ExitGames.Client.Photon.ErrorCode.OperationNotAllowedInCurrentState = -3`

(-3) Operation can't be executed yet (e.g.

OpJoin can't be called before being authenticated, RaiseEvent can't be used before getting into a room).

Before you call any operations on the Cloud servers, the automated client workflow must complete its authorization. In PUN, wait until State is: JoinedLobby (with AutoJoinLobby = true) or ConnectedToMaster (AutoJoinLobby = false)

8.6.2.17 `const int ExitGames.Client.Photon.ErrorCode.PluginMismatch = 0x7FFF - 16`

(32751) CreateGame/JoinGame/Join operation fails if expected plugin does not correspond to loaded one.

8.6.2.18 `const int ExitGames.Client.Photon.ErrorCode.PluginReportedError = 0x7FFF - 15`

(32752) A server-side plugin (or webhook) failed to execute and reported an error.

Check the OperationResponse.DebugMessage.

8.6.2.19 `const int ExitGames.Client.Photon.ErrorCode.ServerFull = 0x7FFF - 5`

(32762) Not in use currently.

8.6.2.20 `const int ExitGames.Client.Photon.ErrorCode.UserBlocked = 0x7FFF - 6`

(32761) Not in use currently.

The documentation for this class was generated from the following file:

- `C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/Photon↔ Network/LoadbalancingPeer.cs`

8.7 `ExitGames.Client.Photon.EventCode` Class Reference

Class for constants.

Public Attributes

- `const byte GameList = 230`
(230) Initial list of RoomInfos (in lobby on Master)
- `const byte GameListUpdate = 229`
(229) Update of RoomInfos to be merged into "initial" list (in lobby on Master)
- `const byte QueueState = 228`
(228) Currently not used. State of queueing in case of server-full
- `const byte Match = 227`
(227) Currently not used. Event for matchmaking
- `const byte AppStats = 226`
(226) Event with stats about this application (players, rooms, etc)
- `const byte LobbyStats = 224`
(224) This event provides a list of lobbies with their player and game counts.
- `const byte AzureNodeInfo = 210`
(210) Internally used in case of hosting by Azure
- `const byte Join = (byte)255`
(255) Event Join: someone joined the game. The new actorNumber is provided as well as the properties of that actor (if set in OpJoin).
- `const byte Leave = (byte)254`
(254) Event Leave: The player who left the game can be identified by the actorNumber.
- `const byte PropertiesChanged = (byte)253`
(253) When you call OpSetProperties with the broadcast option "on", this event is fired. It contains the properties being set.
- `const byte SetPropertyies = (byte)253`
(253) When you call OpSetProperties with the broadcast option "on", this event is fired. It contains the properties being set.
- `const byte ErrorInfo = 251`
(252) When player left game unexpected and the room has a playerTtl > 0, this event is fired to let everyone know about the timeout.
- `const byte CacheSliceChanged = 250`
(250) Sent by Photon when the event cache slice was changed. Done by OpRaiseEvent.

8.7.1 Detailed Description

Class for constants.

These values are for events defined by [Photon](#) Loadbalancing. Pun uses these constants internally.

They start at 255 and go DOWN. Your own in-game events can start at 0.

8.7.2 Member Data Documentation

8.7.2.1 `const byte ExitGames.Client.Photon.EventCode.AppStats = 226`

(226) Event with stats about this application (players, rooms, etc)

8.7.2.2 `const byte ExitGames.Client.Photon.EventCode.AzureNodeInfo = 210`

(210) Internally used in case of hosting by Azure

8.7.2.3 `const byte ExitGames.Client.Photon.EventCode.CacheSliceChanged = 250`

(250) Sent by [Photon](#) when the event cache slice was changed. Done by `OpRaiseEvent`.

8.7.2.4 `const byte ExitGames.Client.Photon.EventCode.ErrorInfo = 251`

(252) When player left game unexpected and the room has a `playerTtl > 0`, this event is fired to let everyone know about the timeout.

Obsolete. Replaced by `Leave`. `public const byte Disconnect = LiteEventCode.Disconnect;`

(251) Sent by [Photon](#) Cloud when a plugin-call failed. Usually, the execution on the server continues, despite the issue. Contains: [ParameterCode.Info](#).

8.7.2.5 `const byte ExitGames.Client.Photon.EventCode.GameList = 230`

(230) Initial list of `RoomInfos` (in lobby on Master)

8.7.2.6 `const byte ExitGames.Client.Photon.EventCode.GameListUpdate = 229`

(229) Update of `RoomInfos` to be merged into "initial" list (in lobby on Master)

8.7.2.7 `const byte ExitGames.Client.Photon.EventCode.Join = (byte)255`

(255) Event Join: someone joined the game. The new `actorNumber` is provided as well as the properties of that actor (if set in `OpJoin`).

8.7.2.8 `const byte ExitGames.Client.Photon.EventCode.Leave = (byte)254`

(254) Event Leave: The player who left the game can be identified by the actorNumber.

8.7.2.9 `const byte ExitGames.Client.Photon.EventCode.LobbyStats = 224`

(224) This event provides a list of lobbies with their player and game counts.

8.7.2.10 `const byte ExitGames.Client.Photon.EventCode.Match = 227`

(227) Currently not used. Event for matchmaking

8.7.2.11 `const byte ExitGames.Client.Photon.EventCode.PropertiesChanged = (byte)253`

(253) When you call `OpSetProperties` with the broadcast option "on", this event is fired. It contains the properties being set.

8.7.2.12 `const byte ExitGames.Client.Photon.EventCode.QueueState = 228`

(228) Currently not used. State of queueing in case of server-full

8.7.2.13 `const byte ExitGames.Client.Photon.EventCode.SetProperties = (byte)253`

(253) When you call `OpSetProperties` with the broadcast option "on", this event is fired. It contains the properties being set.

The documentation for this class was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/Photon↔ Network/[LoadbalancingPeer.cs](#)

8.8 Extensions Class Reference

This static class defines some useful extension methods for several existing classes (e.g.

Static Public Member Functions

- static [PhotonView\[\] GetPhotonViewsInChildren](#) (this UnityEngine.GameObject go)
- static [PhotonView GetPhotonView](#) (this UnityEngine.GameObject go)
- static bool [AlmostEquals](#) (this Vector3 target, Vector3 second, float sqrMagnitudePrecision)
compares the squared magnitude of target - second to given float value
- static bool [AlmostEquals](#) (this Vector2 target, Vector2 second, float sqrMagnitudePrecision)
compares the squared magnitude of target - second to given float value
- static bool [AlmostEquals](#) (this Quaternion target, Quaternion second, float maxAngle)
compares the angle between target and second to given float value
- static bool [AlmostEquals](#) (this float target, float second, float floatDiff)
compares two floats and returns true if their difference is less than floatDiff
- static void [Merge](#) (this IDictionary target, IDictionary addHash)
Merges all keys from addHash into the target.
- static void [MergeStringKeys](#) (this IDictionary target, IDictionary addHash)
Merges keys of type string to target Hashtable.
- static string [ToStringFull](#) (this IDictionary origin)
Returns a string-representation of the IDictionary's content, including type-information.
- static [Hashtable StripToStringKeys](#) (this IDictionary original)
This method copies all string-typed keys of the original into a new Hashtable.
- static void [StripKeysWithNullValues](#) (this IDictionary original)
This removes all key-value pairs that have a null-reference as value.
- static bool [Contains](#) (this int[] target, int nr)
Checks if a particular integer value is in an int-array.

8.8.1 Detailed Description

This static class defines some useful extension methods for several existing classes (e.g.

Vector3, float and others).

8.8.2 Member Function Documentation

8.8.2.1 static bool Extensions.AlmostEquals (this Vector3 *target*, Vector3 *second*, float *sqrMagnitudePrecision*)
[static]

compares the squared magnitude of target - second to given float value

8.8.2.2 static bool Extensions.AlmostEquals (this Vector2 *target*, Vector2 *second*, float *sqrMagnitudePrecision*)
[static]

compares the squared magnitude of target - second to given float value

8.8.2.3 static bool Extensions.AlmostEquals (this Quaternion *target*, Quaternion *second*, float *maxAngle*) [static]

compares the angle between target and second to given float value

8.8.2.4 static bool Extensions.AlmostEquals (this float *target*, float *second*, float *floatDiff*) [static]

compares two floats and returns true if their difference is less than floatDiff

8.8.2.5 static bool Extensions.Contains (this int[] *target*, int *nr*) [static]

Checks if a particular integer value is in an int-array.

This might be useful to look up if a particular actorNumber is in the list of players of a room.

Parameters

<i>target</i>	The array of ints to check.
<i>nr</i>	The number to lookup in target.

Returns

True if nr was found in target.

8.8.2.6 static PhotonView Extensions.GetPhotonView (this UnityEngine.GameObject *go*) [static]

8.8.2.7 static PhotonView[] Extensions.GetPhotonViewsInChildren (this UnityEngine.GameObject *go*) [static]

8.8.2.8 static void Extensions.Merge (this IDictionary *target*, IDictionary *addHash*) [static]

Merges all keys from addHash into the target.

Adds new keys and updates the values of existing keys in target.

Parameters

<i>target</i>	The IDictionary to update.
<i>addHash</i>	The IDictionary containing data to merge into target.

8.8.2.9 static void Extensions.MergeStringKeys (this IDictionary *target*, IDictionary *addHash*) [static]

Merges keys of type string to target Hashtable.

Does not remove keys from target (so non-string keys CAN be in target if they were before).

Parameters

<i>target</i>	The target IDictionary passed in plus all string-typed keys from the addHash.
<i>addHash</i>	A IDictionary that should be merged partly into target to update it.

8.8.2.10 static void Extensions.StripKeysWithNullValues (this IDictionary *original*) [static]

This removes all key-value pairs that have a null-reference as value.

[Photon](#) properties are removed by setting their value to null. Changes the original passed IDictionary!

Parameters

<i>original</i>	The IDictionary to strip of keys with null-values.
-----------------	--

8.8.2.11 static Hashtable Extensions.StripToStringKeys (this IDictionary *original*) [static]

This method copies all string-typed keys of the original into a new Hashtable.

Does not recurse (!) into hashes that might be values in the root-hash. This does not modify the original.

Parameters

<i>original</i>	The original IDictionary to get string-typed keys from.
-----------------	---

Returns

New Hashtable containing only string-typed keys of the original.

8.8.2.12 static string Extensions.ToStringFull (this IDictionary *origin*) [static]

Returns a string-representation of the IDictionary's content, including type-information.

Note: This might turn out a "heavy-duty" call if used frequently but it's usfuly to debug Dictionary or Hashtable content.

Parameters

<i>origin</i>	Some Dictionary or Hashtable.
---------------	-------------------------------

Returns

String of the content of the IDictionary.

The documentation for this class was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/Photon↔ Network/[Extensions.cs](#)

8.9 FriendInfo Class Reference

Used to store info about a friend's online state and in which room he/she is.

Public Member Functions

- override string [ToString](#) ()

Properties

- string [Name](#) [get, protected set]
- bool [IsOnline](#) [get, protected set]
- string [Room](#) [get, protected set]
- bool [IsInRoom](#) [get]

8.9.1 Detailed Description

Used to store info about a friend's online state and in which room he/she is.

8.9.2 Member Function Documentation

8.9.2.1 override string [FriendInfo.ToString](#) ()

8.9.3 Property Documentation

8.9.3.1 bool [FriendInfo.IsInRoom](#) [get]

8.9.3.2 bool [FriendInfo.IsOnline](#) [get],[protected set]

8.9.3.3 string [FriendInfo.Name](#) [get],[protected set]

8.9.3.4 string [FriendInfo.Room](#) [get],[protected set]

The documentation for this class was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/Photon↔ Network/[FriendInfo.cs](#)

8.10 GameObjectExtensions Class Reference

Small number of extension methods that make it easier for PUN to work cross-Unity-versions.

Static Public Member Functions

- static bool [GetActive](#) (this GameObject target)
Unity-version-independent replacement for active GO property.

8.10.1 Detailed Description

Small number of extension methods that make it easier for PUN to work cross-Unity-versions.

8.10.2 Member Function Documentation

8.10.2.1 `static bool GameObjectExtensions.GetActive (this GameObject target) [static]`

Unity-version-independent replacement for active GO property.

Returns

Unity 3.5: active. Any newer Unity: activeInHierarchy.

The documentation for this class was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/Photon↔ Network/[Extensions.cs](#)

8.11 ExitGames.Client.Photon.GamePropertyKey Class Reference

Class for constants.

Public Attributes

- const byte [MaxPlayers](#) = 255
(255) Max number of players that "fit" into this room. 0 is for "unlimited".
- const byte [IsVisible](#) = 254
(254) Makes this room listed or not in the lobby on master.
- const byte [IsOpen](#) = 253
(253) Allows more players to join a room (or not).
- const byte [PlayerCount](#) = 252
(252) Current count of players in the room. Used only in the lobby on master.
- const byte [Removed](#) = 251
(251) True if the room is to be removed from room listing (used in update to room list in lobby on master)
- const byte [PropsListedInLobby](#) = 250
(250) A list of the room properties to pass to the [RoomInfo](#) list in a lobby. This is used in `CreateRoom`, which defines this list once per room.
- const byte [CleanupCacheOnLeave](#) = 249
(249) Equivalent of Operation Join parameter `CleanupCacheOnLeave`.
- const byte [MasterClientId](#) = (byte)248
(248) Code for MasterClientId, which is synced by server.

8.11.1 Detailed Description

Class for constants.

These (byte) values are for "well known" room/game properties used in [Photon](#) Loadbalancing. Pun uses these constants internally.

"Custom properties" have to use a string-type as key. They can be assigned at will.

8.11.2 Member Data Documentation

8.11.2.1 `const byte ExitGames.Client.Photon.GamePropertyKey.CleanupCacheOnLeave = 249`

(249) Equivalent of Operation Join parameter CleanupCacheOnLeave.

8.11.2.2 `const byte ExitGames.Client.Photon.GamePropertyKey.IsOpen = 253`

(253) Allows more players to join a room (or not).

8.11.2.3 `const byte ExitGames.Client.Photon.GamePropertyKey.IsVisible = 254`

(254) Makes this room listed or not in the lobby on master.

8.11.2.4 `const byte ExitGames.Client.Photon.GamePropertyKey.MasterClientId = (byte)248`

(248) Code for MasterClientId, which is synced by server.

When sent as op-parameter this is (byte)203. As room property this is (byte)248.

Tightly related to [ParameterCode.MasterClientId](#).

8.11.2.5 `const byte ExitGames.Client.Photon.GamePropertyKey.MaxPlayers = 255`

(255) Max number of players that "fit" into this room. 0 is for "unlimited".

8.11.2.6 `const byte ExitGames.Client.Photon.GamePropertyKey.PlayerCount = 252`

(252) Current count of players in the room. Used only in the lobby on master.

8.11.2.7 `const byte ExitGames.Client.Photon.GamePropertyKey.PropsListedInLobby = 250`

(250) A list of the room properties to pass to the [RoomInfo](#) list in a lobby. This is used in CreateRoom, which defines this list once per room.

8.11.2.8 `const byte ExitGames.Client.Photon.GamePropertyKey.Removed = 251`

(251) True if the room is to be removed from room listing (used in update to room list in lobby on master)

The documentation for this class was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/Photon↵ Network/[LoadbalancingPeer.cs](#)

8.12 ExitGames.Client.GUI.GizmoTypeDrawer Class Reference

Static Public Member Functions

- static void [Draw](#) (Vector3 center, [GizmoType](#) type, Color color, float size)

8.12.1 Member Function Documentation

8.12.1.1 `static void ExitGames.Client.GUI.GizmoTypeDrawer.Draw (Vector3 center, GizmoType type, Color color, float size)`
[static]

The documentation for this class was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/Photon↵ Network/[GizmoType.cs](#)

8.13 HelpURL Class Reference

Empty implementation of the upcoming [HelpURL](#) of Unity 5.1.

Inherits [Attribute](#).

Public Member Functions

- [HelpURL](#) (string url)

8.13.1 Detailed Description

Empty implementation of the upcoming [HelpURL](#) of Unity 5.1.

This one is only for compatibility of attributes.

<http://feedback.unity3d.com/suggestions/override-component-documentation-slash-help-link>

8.13.2 Constructor & Destructor Documentation

8.13.2.1 `HelpURL.HelpURL (string url)`

The documentation for this class was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/Photon↔ Network/[PhotonClasses.cs](#)

8.14 HighlightOwnedGameObj Class Reference

Inherits [Photon.MonoBehaviour](#).

Public Attributes

- GameObject [PointerPrefab](#)
- float [Offset](#) = 1.5f

Additional Inherited Members

8.14.1 Member Data Documentation

8.14.1.1 float `HighlightOwnedGameObj.Offset` = 1.5f

8.14.1.2 GameObject `HighlightOwnedGameObj.PointerPrefab`

The documentation for this class was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/UtilityScripts/[Highlight↔ OwnedGameObj.cs](#)

8.15 InputToEvent Class Reference

Utility component to forward mouse or touch input to clicked gameobjects.

Inherits `MonoBehaviour`.

Public Attributes

- bool [DetectPointedAtGameObject](#)
- bool [Dragging](#)

Static Public Attributes

- static Vector3 [inputHitPos](#)

Properties

- static GameObject [goPointedAt](#) [get]
- Vector2 [DragVector](#) [get]

8.15.1 Detailed Description

Utility component to forward mouse or touch input to clicked gameobjects.

Calls OnPress, OnClick and OnRelease methods on "first" game object.

8.15.2 Member Data Documentation

8.15.2.1 bool [InputToEvent.DetectPointedAtGameObject](#)

8.15.2.2 bool [InputToEvent.Dragging](#)

8.15.2.3 Vector3 [InputToEvent.inputHitPos](#) [static]

8.15.3 Property Documentation

8.15.3.1 Vector2 [InputToEvent.DragVector](#) [get]

8.15.3.2 GameObject [InputToEvent.goPointedAt](#) [static], [get]

The documentation for this class was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/UtilityScripts/[InputToEvent.cs](#)

8.16 InRoomChat Class Reference

Inherits [Photon.MonoBehaviour](#).

Public Member Functions

- void [Start](#) ()
- void [OnGUI](#) ()
- void [Chat](#) (string newLine, [PhotonMessageInfo](#) mi)
- void [AddLine](#) (string newLine)

Public Attributes

- Rect [GuiRect](#) = new Rect(0,0, 250,250)
- bool [IsVisible](#) = true
- bool [AlignBottom](#) = false
- List< string > [messages](#) = new List<string>()

Static Public Attributes

- static readonly string [ChatRPC](#) = "Chat"

Additional Inherited Members

8.16.1 Member Function Documentation

8.16.1.1 void InRoomChat.AddLine (string *newLine*)

8.16.1.2 void InRoomChat.Chat (string *newLine*, PhotonMessageInfo *mi*)

8.16.1.3 void InRoomChat.OnGUI ()

8.16.1.4 void InRoomChat.Start ()

8.16.2 Member Data Documentation

8.16.2.1 bool InRoomChat.AlignBottom = false

8.16.2.2 readonly string InRoomChat.ChatRPC = "Chat" [static]

8.16.2.3 Rect InRoomChat.GuiRect = new Rect(0,0, 250,250)

8.16.2.4 bool InRoomChat.IsVisible = true

8.16.2.5 List<string> InRoomChat.messages = new List<string>()

The documentation for this class was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/UtilityScripts/[InRoomChat.cs](#)

8.17 InRoomRoundTimer Class Reference

Simple script that uses a property to sync a start time for a multiplayer game.

Inherits MonoBehaviour.

Public Member Functions

- void `OnJoinedRoom` ()
Called by PUN when this client entered a room (no matter if joined or created).
- void `OnPhotonCustomRoomPropertiesChanged` (`Hashtable` `propertiesThatChanged`)
Called by PUN when new properties for the room were set (by any client in the room).
- void `OnMasterClientSwitched` (`PhotonPlayer` `newMasterClient`)
- void `OnGUI` ()

Public Attributes

- int `SecondsPerTurn` = 500
- double `StartTime`
- Rect `TextPos` = new Rect(0,80,150,300)
- double `starting_severtime`
- double `timetostart` = 0.0f
- double `secondsbeforeend` = 0

8.17.1 Detailed Description

Simple script that uses a property to sync a start time for a multiplayer game.

When entering a room, the first player will store the synchronized timestamp. You can't set the room's synchronized time in `CreateRoom`, because the clock on the Master Server and those on the Game Servers are not in sync. We use many servers and each has it's own timer.

Everyone else will join the room and check the property to calculate how much time passed since start. You can start a new round whenever you like.

Based on this, you should be able to implement a synchronized timer for turns between players.

8.17.2 Member Function Documentation

8.17.2.1 void InRoomRoundTimer.OnGUI ()

8.17.2.2 void InRoomRoundTimer.OnJoinedRoom ()

Called by PUN when this client entered a room (no matter if joined or created).

8.17.2.3 void InRoomRoundTimer.OnMasterClientSwitched (`PhotonPlayer` *newMasterClient*)

In theory, the client which created the room might crash/close before it sets the start time. Just to make extremely sure this never happens, a new masterClient will check if it has to start a new round.

8.17.2.4 void InRoomRoundTimer.OnPhotonCustomRoomPropertiesChanged (`Hashtable` *propertiesThatChanged*)

Called by PUN when new properties for the room were set (by any client in the room).

8.17.3 Member Data Documentation

8.17.3.1 `double InRoomRoundTimer.secondsbeforeend = 0`

8.17.3.2 `int InRoomRoundTimer.SecondsPerTurn = 500`

8.17.3.3 `double InRoomRoundTimer.starting_severtime`

8.17.3.4 `double InRoomRoundTimer.StartTime`

8.17.3.5 `Rect InRoomRoundTimer.TextPos = new Rect(0,80,150,300)`

8.17.3.6 `double InRoomRoundTimer.timetostart = 0.0f`

The documentation for this class was generated from the following file:

- [C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/UtilityScripts/InRoomRoundTimer.cs](#)

8.18 IPunCallbacks Interface Reference

This interface is used as definition of all callback methods of PUN, except `OnPhotonSerializeView`.

Inherited by [Photon.PunBehaviour](#).

Public Member Functions

- void [OnConnectedToPhoton](#) ()
Called when the initial connection got established but before you can use the server.
- void [OnLeftRoom](#) ()
Called when the local user/client left a room.
- void [OnMasterClientSwitched](#) ([PhotonPlayer](#) newMasterClient)
Called after switching to a new MasterClient when the current one leaves.
- void [OnPhotonCreateRoomFailed](#) (object[] codeAndMsg)
Called when a `CreateRoom()` call failed.
- void [OnPhotonJoinRoomFailed](#) (object[] codeAndMsg)
Called when a `JoinRoom()` call failed.
- void [OnCreatedRoom](#) ()
Called when this client created a room and entered it.
- void [OnJoinedLobby](#) ()
Called on entering a lobby on the Master Server.
- void [OnLeftLobby](#) ()
Called after leaving a lobby.
- void [OnFailedToConnectToPhoton](#) ([DisconnectCause](#) cause)
Called if a connect call to the [Photon](#) server failed before the connection was established, followed by a call to [OnDisconnectedFromPhoton\(\)](#).
- void [OnConnectionFail](#) ([DisconnectCause](#) cause)

- Called when something causes the connection to fail (after it was established), followed by a call to [OnDisconnectedFromPhoton\(\)](#).*
- void [OnDisconnectedFromPhoton](#) ()
Called after disconnecting from the [Photon](#) server.
 - void [OnPhotonInstantiate](#) ([PhotonMessageInfo](#) info)
Called on all scripts on a [GameObject](#) (and children) that have been Instantiated using [PhotonNetwork.Instantiate](#).
 - void [OnReceivedRoomListUpdate](#) ()
Called for any update of the room-listing while in a lobby ([PhotonNetwork.insideLobby](#)) on the Master Server.
 - void [OnJoinedRoom](#) ()
Called when entering a room (by creating or joining it).
 - void [OnPhotonPlayerConnected](#) ([PhotonPlayer](#) newPlayer)
Called when a remote player entered the room.
 - void [OnPhotonPlayerDisconnected](#) ([PhotonPlayer](#) otherPlayer)
Called when a remote player left the room.
 - void [OnPhotonRandomJoinFailed](#) (object[] codeAndMsg)
Called when a [JoinRandom\(\)](#) call failed.
 - void [OnConnectedToMaster](#) ()
Called after the connection to the master is established and authenticated but only when [PhotonNetwork.autoJoinLobby](#) is false.
 - void [OnPhotonMaxCccuReached](#) ()
Because the concurrent user limit was (temporarily) reached, this client is rejected by the server and disconnecting.
 - void [OnPhotonCustomRoomPropertiesChanged](#) ([Hashtable](#) propertiesThatChanged)
Called when a room's custom properties changed.
 - void [OnPhotonPlayerPropertiesChanged](#) (object[] playerAndUpdatedProps)
Called when custom player-properties are changed.
 - void [OnUpdatedFriendList](#) ()
Called when the server sent the response to a [FindFriends](#) request and updated [PhotonNetwork.Friends](#).
 - void [OnCustomAuthenticationFailed](#) (string debugMessage)
Called when the custom authentication failed.
 - void [OnWebRpcResponse](#) ([OperationResponse](#) response)
Called by PUN when the response to a [WebRPC](#) is available.
 - void [OnOwnershipRequest](#) (object[] viewAndPlayer)
Called when another player requests ownership of a [PhotonView](#) from you (the current owner).
 - void [OnLobbyStatisticsUpdate](#) ()
Called when the Master Server sent an update for the Lobby Statistics, updating [PhotonNetwork.LobbyStatistics](#).

8.18.1 Detailed Description

This interface is used as definition of all callback methods of PUN, except [OnPhotonSerializeView](#).

Preferably, implement them individually.

This interface is available for completeness, more than for actually implementing it in a game. You can implement each method individually in any [MonoBehaviour](#), without implementing [IPunCallbacks](#).

PUN calls all callbacks by name. Don't use implement callbacks with fully qualified name. Example: [IPunCallbacks.OnConnectedToPhoton](#) won't get called by Unity's [SendMessage\(\)](#).

PUN will call these methods on any script that implements them, analog to Unity's events and callbacks. The situation that triggers the call is described per method.

[OnPhotonSerializeView](#) is NOT called like these callbacks! It's usage frequency is much higher and it is implemented in: [IPunObservable](#).

8.18.2 Member Function Documentation

8.18.2.1 void IPunCallbacks.OnConnectedToMaster ()

Called after the connection to the master is established and authenticated but only when [PhotonNetwork.autoJoinLobby](#) is false.

If you set [PhotonNetwork.autoJoinLobby](#) to true, [OnJoinedLobby\(\)](#) will be called instead of this.

You can join rooms and create them even without being in a lobby. The default lobby is used in that case. The list of available rooms won't become available unless you join a lobby via [PhotonNetwork.joinLobby](#).

Implemented in [Photon.PunBehaviour](#).

8.18.2.2 void IPunCallbacks.OnConnectedToPhoton ()

Called when the initial connection got established but before you can use the server.

[OnJoinedLobby\(\)](#) or [OnConnectedToMaster\(\)](#) are called when PUN is ready.

This callback is only useful to detect if the server can be reached at all (technically). Most often, it's enough to implement [OnFailedToConnectToPhoton\(\)](#) and [OnDisconnectedFromPhoton\(\)](#).

[OnJoinedLobby\(\)](#) or [OnConnectedToMaster\(\)](#) are called when PUN is ready.

When this is called, the low level connection is established and PUN will send your AppId, the user, etc in the background. This is not called for transitions from the masterserver to game servers.

Implemented in [Photon.PunBehaviour](#).

8.18.2.3 void IPunCallbacks.OnConnectionFail (DisconnectCause cause)

Called when something causes the connection to fail (after it was established), followed by a call to [OnDisconnectedFromPhoton\(\)](#).

If the server could not be reached in the first place, [OnFailedToConnectToPhoton](#) is called instead. The reason for the error is provided as [DisconnectCause](#).

Implemented in [Photon.PunBehaviour](#).

8.18.2.4 void IPunCallbacks.OnCreatedRoom ()

Called when this client created a room and entered it.

[OnJoinedRoom\(\)](#) will be called as well.

This callback is only called on the client which created a room (see [PhotonNetwork.CreateRoom](#)).

As any client might close (or drop connection) anytime, there is a chance that the creator of a room does not execute [OnCreatedRoom](#).

If you need specific room properties or a "start signal", it is safer to implement [OnMasterClientSwitched\(\)](#) and to make the new MasterClient check the room's state.

Implemented in [Photon.PunBehaviour](#).

8.18.2.5 void IPunCallbacks.OnCustomAuthenticationFailed (string *debugMessage*)

Called when the custom authentication failed.

Followed by disconnect!

Custom Authentication can fail due to user-input, bad tokens/secrets. If authentication is successful, this method is not called. Implement [OnJoinedLobby\(\)](#) or [OnConnectedToMaster\(\)](#) (as usual).

During development of a game, it might also fail due to wrong configuration on the server side. In those cases, logging the *debugMessage* is very important.

Unless you setup a custom authentication service for your app (in the [Dashboard](#)), this won't be called!

Parameters

<i>debugMessage</i>	Contains a debug message why authentication failed. This has to be fixed during development time.
---------------------	---

Implemented in [Photon.PunBehaviour](#).

8.18.2.6 void IPunCallbacks.OnDisconnectedFromPhoton ()

Called after disconnecting from the [Photon](#) server.

In some cases, other callbacks are called before `OnDisconnectedFromPhoton` is called. Examples: [OnConnectionFail\(\)](#) and [OnFailedToConnectToPhoton\(\)](#).

Implemented in [Photon.PunBehaviour](#).

8.18.2.7 void IPunCallbacks.OnFailedToConnectToPhoton (DisconnectCause cause)

Called if a connect call to the [Photon](#) server failed before the connection was established, followed by a call to [OnDisconnectedFromPhoton\(\)](#).

This is called when no connection could be established at all. It differs from `OnConnectionFail`, which is called when an existing connection fails.

Implemented in [Photon.PunBehaviour](#).

8.18.2.8 void IPunCallbacks.OnJoinedLobby ()

Called on entering a lobby on the Master Server.

The actual room-list updates will call [OnReceivedRoomListUpdate\(\)](#).

Note: When [PhotonNetwork.autoJoinLobby](#) is false, [OnConnectedToMaster\(\)](#) will be called and the room list won't become available.

While in the lobby, the roomlist is automatically updated in fixed intervals (which you can't modify). The room list gets available when [OnReceivedRoomListUpdate\(\)](#) gets called after [OnJoinedLobby\(\)](#).

Implemented in [Photon.PunBehaviour](#).

8.18.2.9 void IPunCallbacks.OnJoinedRoom ()

Called when entering a room (by creating or joining it).

Called on all clients (including the Master Client).

This method is commonly used to instantiate player characters. If a match has to be started "actively", you can call an [PunRPC](#) triggered by a user's button-press or a timer.

When this is called, you can usually already access the existing players in the room via [PhotonNetwork.playerList](#). Also, all custom properties should be already available as [Room.customProperties](#). Check [Room.playerCount](#) to find out if enough players are in the room to start playing.

Implemented in [Photon.PunBehaviour](#).

8.18.2.10 void IPunCallbacks.OnLeftLobby ()

Called after leaving a lobby.

When you leave a lobby, [CreateRoom](#) and [JoinRandomRoom](#) automatically refer to the default lobby.

Implemented in [Photon.PunBehaviour](#).

8.18.2.11 void IPunCallbacks.OnLeftRoom ()

Called when the local user/client left a room.

When leaving a room, PUN brings you back to the Master Server. Before you can use lobbies and join or create rooms, [OnJoinedLobby\(\)](#) or [OnConnectedToMaster\(\)](#) will get called again.

Implemented in [Photon.PunBehaviour](#).

8.18.2.12 void IPunCallbacks.OnLobbyStatisticsUpdate ()

Called when the Master Server sent an update for the Lobby Statistics, updating [PhotonNetwork.LobbyStatistics](#).

This callback has two preconditions: `EnableLobbyStatistics` must be set to true, before this client connects. And the client has to be connected to the Master Server, which is providing the info about lobbies.

Implemented in [Photon.PunBehaviour](#).

8.18.2.13 void IPunCallbacks.OnMasterClientSwitched (PhotonPlayer newMasterClient)

Called after switching to a new MasterClient when the current one leaves.

This is not called when this client enters a room. The former MasterClient is still in the player list when this method get called.

Implemented in [Photon.PunBehaviour](#).

8.18.2.14 void IPunCallbacks.OnOwnershipRequest (object[] viewAndPlayer)

Called when another player requests ownership of a [PhotonView](#) from you (the current owner).

The parameter `viewAndPlayer` contains:

[PhotonView](#) `view` = `viewAndPlayer[0]` as [PhotonView](#);

[PhotonPlayer](#) `requestingPlayer` = `viewAndPlayer[1]` as [PhotonPlayer](#);

Parameters

<i>viewAndPlayer</i>	The PhotonView is <code>viewAndPlayer[0]</code> and the requesting player is <code>viewAndPlayer[1]</code> .
----------------------	--

Implemented in [Photon.PunBehaviour](#).

8.18.2.15 void IPunCallbacks.OnPhotonCreateRoomFailed (object[] *codeAndMsg*)

Called when a CreateRoom() call failed.

The parameter provides ErrorCode and message (as array).

Most likely because the room name is already in use (some other client was faster than you). PUN logs some info if the [PhotonNetwork.logLevel](#) is \geq [PhotonLogLevel.Informational](#).

Parameters

<i>codeAndMsg</i>	<i>codeAndMsg</i> [0] is short ErrorCode and <i>codeAndMsg</i> [1] is a string debug msg.
-------------------	---

Implemented in [Photon.PunBehaviour](#).

8.18.2.16 void IPunCallbacks.OnPhotonCustomRoomPropertiesChanged (Hashtable *propertiesThatChanged*)

Called when a room's custom properties changed.

The *propertiesThatChanged* contains all that was set via [Room.SetCustomProperties](#).

Since v1.25 this method has one parameter: Hashtable *propertiesThatChanged*.
Changing properties must be done by [Room.SetCustomProperties](#), which causes this callback locally, too.

Parameters

<i>propertiesThatChanged</i>	
------------------------------	--

Implemented in [Photon.PunBehaviour](#).

8.18.2.17 void IPunCallbacks.OnPhotonInstantiate (PhotonMessageInfo *info*)

Called on all scripts on a GameObject (and children) that have been Instantiated using [PhotonNetwork.Instantiate](#).

[PhotonMessageInfo](#) parameter provides info about who created the object and when (based off Photon↔ Networking.time).

Implemented in [Photon.PunBehaviour](#).

8.18.2.18 void IPunCallbacks.OnPhotonJoinRoomFailed (object[] *codeAndMsg*)

Called when a JoinRoom() call failed.

The parameter provides ErrorCode and message (as array).

Most likely error is that the room does not exist or the room is full (some other client was faster than you). PUN logs some info if the [PhotonNetwork.logLevel](#) is \geq [PhotonLogLevel.Informational](#).

Parameters

<i>codeAndMsg</i>	codeAndMsg[0] is short ErrorCode and codeAndMsg[1] is string debug msg.
-------------------	---

Implemented in [Photon.PunBehaviour](#).

8.18.2.19 void IPunCallbacks.OnPhotonMaxCcuReached ()

Because the concurrent user limit was (temporarily) reached, this client is rejected by the server and disconnecting. When this happens, the user might try again later. You can't create or join rooms in OnPhotonMaxCcuReached(), cause the client will be disconnecting. You can raise the CCU limits with a new license (when you host yourself) or extended subscription (when using the [Photon](#) Cloud). The [Photon](#) Cloud will mail you when the CCU limit was reached. This is also visible in the Dashboard (webpage).

Implemented in [Photon.PunBehaviour](#).

8.18.2.20 void IPunCallbacks.OnPhotonPlayerConnected ([PhotonPlayer](#) *newPlayer*)

Called when a remote player entered the room.

This [PhotonPlayer](#) is already added to the playerlist at this time.

If your game starts with a certain number of players, this callback can be useful to check the [Room.playerCount](#) and find out if you can start.

Implemented in [Photon.PunBehaviour](#).

8.18.2.21 void IPunCallbacks.OnPhotonPlayerDisconnected ([PhotonPlayer](#) *otherPlayer*)

Called when a remote player left the room.

This [PhotonPlayer](#) is already removed from the playerlist at this time.

When your client calls PhotonNetwork.leaveRoom, PUN will call this method on the remaining clients. When a remote client drops connection or gets closed, this callback gets executed. after a timeout of several seconds.

Implemented in [Photon.PunBehaviour](#).

8.18.2.22 void IPunCallbacks.OnPhotonPlayerPropertiesChanged (object[] *playerAndUpdatedProps*)

Called when custom player-properties are changed.

Player and the changed properties are passed as object[].

Since v1.25 this method has one parameter: object[] playerAndUpdatedProps, which contains two entries.

[0] is the affected [PhotonPlayer](#).

[1] is the Hashtable of properties that changed.

We are using a object[] due to limitations of Unity's GameObject.SendMessage (which has only one optional parameter).

Changing properties must be done by [PhotonPlayer.SetCustomProperties](#), which causes this callback locally, too.

Example:

```
void OnPhotonPlayerPropertiesChanged(object[] playerAndUpdatedProps) {
    PhotonPlayer player = playerAndUpdatedProps[0] as PhotonPlayer;
    Hashtable props = playerAndUpdatedProps[1] as Hashtable;
    //...
}
```

Parameters

<i>playerAndUpdatedProps</i>	Contains PhotonPlayer and the properties that changed See remarks.
------------------------------	--

Implemented in [Photon.PunBehaviour](#).

8.18.2.23 void IPunCallbacks.OnPhotonRandomJoinFailed (object[] *codeAndMsg*)

Called when a JoinRandom() call failed.

The parameter provides ErrorCode and message.

Most likely all rooms are full or no rooms are available.

When using multiple lobbies (via JoinLobby or [TypedLobby](#)), another lobby might have more/fitting rooms.

PUN logs some info if the [PhotonNetwork.logLevel](#) is \geq [PhotonLogLevel.Informational](#).

Parameters

<i>codeAndMsg</i>	codeAndMsg[0] is short ErrorCode. codeAndMsg[1] is string debug msg.
-------------------	--

Implemented in [Photon.PunBehaviour](#).

8.18.2.24 void IPunCallbacks.OnReceivedRoomListUpdate ()

Called for any update of the room-listing while in a lobby ([PhotonNetwork.insideLobby](#)) on the Master Server.

PUN provides the list of rooms by [PhotonNetwork.GetRoomList\(\)](#).

Each item is a [RoomInfo](#) which might include custom properties (provided you defined those as lobby-listed when creating a room).

Not all types of lobbies provide a listing of rooms to the client. Some are silent and specialized for server-side matchmaking.

Implemented in [Photon.PunBehaviour](#).

8.18.2.25 void IPunCallbacks.OnUpdatedFriendList ()

Called when the server sent the response to a FindFriends request and updated [PhotonNetwork.Friends](#).

The friends list is available as [PhotonNetwork.Friends](#), listing name, online state and the room a user is in (if any).

Implemented in [Photon.PunBehaviour](#).

8.18.2.26 void IPunCallbacks.OnWebRpcResponse (OperationResponse response)

Called by PUN when the response to a WebRPC is available.

See PhotonNetwork.WebRPC.

Important: The response.ReturnCode is 0 if Photon was able to reach your web-service.

The content of the response is what your web-service sent. You can create a WebRpcResponse from it.

Example: WebRpcResponse webResponse = new WebRpcResponse(operationResponse);

Please note: Class OperationResponse is in a namespace which needs to be "used":
using ExitGames.Client.Photon; // includes OperationResponse (and other classes)

The OperationResponse.ReturnCode by Photon is:

```
0 for "OK"
-3 for "Web-Service not configured" (see Dashboard / WebHooks)
-5 for "Web-Service does now have RPC path/name" (at least for Azure)
```

Implemented in Photon.PunBehaviour.

The documentation for this interface was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/Photon↔ Network/PhotonClasses.cs

8.19 IPunObservable Interface Reference

Defines the OnPhotonSerializeView method to make it easy to implement correctly for observable scripts.

Inherited by PhotonTransformView, and PickupItem.

Public Member Functions

- void OnPhotonSerializeView (PhotonStream stream, PhotonMessageInfo info)
Called by PUN several times per second, so that your script can write and read synchronization data for the Photon↔ View.

8.19.1 Detailed Description

Defines the OnPhotonSerializeView method to make it easy to implement correctly for observable scripts.

The documentation for this interface was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/Photon↔ Network/PhotonClasses.cs

8.20 IPunPrefabPool Interface Reference

Defines all the methods that a Object Pool must implement, so that PUN can use it.

Public Member Functions

- GameObject [Instantiate](#) (string prefabId, Vector3 position, Quaternion rotation)
This is called when PUN wants to create a new instance of an entity prefab.
- void [Destroy](#) (GameObject gameObject)
This is called when PUN wants to destroy the instance of an entity prefab.

8.20.1 Detailed Description

Defines all the methods that a Object Pool must implement, so that PUN can use it.

To use a Object Pool for instantiation, you can set PhotonNetwork.ObjectPool. That is used for all objects, as long as ObjectPool is not null. The pool has to return a valid non-null GameObject when PUN calls Instantiate. Also, the position and rotation must be applied.

Please note that pooled GameObjects don't get the usual Awake and Start calls. OnEnable will be called (by your pool) but the networking values are not updated yet when that happens. OnEnable will have outdated values for [PhotonView](#) (isMine, etc.). You might have to adjust scripts.

PUN will call OnPhotonInstantiate (see [IPunCallbacks](#)). This should be used to setup the re-used object with regards to networking values / ownership.

8.20.2 Member Function Documentation

8.20.2.1 void IPunPrefabPool.Destroy (GameObject *gameObject*)

This is called when PUN wants to destroy the instance of an entity prefab.

A pool needs some way to find out which type of GameObject got returned via [Destroy\(\)](#). It could be a tag or name or anything similar.

Parameters

<i>gameObject</i>	The instance to destroy.
-------------------	--------------------------

8.20.2.2 GameObject IPunPrefabPool.Instantiate (string *prefabId*, Vector3 *position*, Quaternion *rotation*)

This is called when PUN wants to create a new instance of an entity prefab.

Must return valid GameObject with [PhotonView](#).

Parameters

<i>prefab</i> ↔ <i>Id</i>	The id of this prefab.
<i>position</i>	The position we want the instance instantiated at.
<i>rotation</i>	The rotation we want the instance to take.

Returns

The newly instantiated object, or null if a prefab with *prefabId* was not found.

The documentation for this interface was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/Photon↔Network/[PhotonClasses.cs](#)

8.21 ManualPhotonViewAllocator Class Reference

Inherits MonoBehaviour.

Public Member Functions

- void [AllocateManualPhotonView](#) ()
- void [InstantiateRpc](#) (int viewID)

Public Attributes

- GameObject [Prefab](#)

8.21.1 Member Function Documentation

8.21.1.1 void ManualPhotonViewAllocator.AllocateManualPhotonView ()

8.21.1.2 void ManualPhotonViewAllocator.InstantiateRpc (int viewID)

8.21.2 Member Data Documentation

8.21.2.1 GameObject ManualPhotonViewAllocator.Prefab

The documentation for this class was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/UtilityScripts/[Manual↔PhotonViewAllocator.cs](#)

8.22 Photon.MonoBehaviour Class Reference

This class adds the property `photonView`, while logging a warning when your game still uses the `networkView`.

Inherits `MonoBehaviour`.

Inherited by [ConnectAndJoinRandom](#), [HighlightOwnedGameObj](#), [InRoomChat](#), [MoveByKeys](#), [OnAwakeUse↵](#) [PhotonView](#), [OnClickDestroy](#), [Photon.PunBehaviour](#), [PhotonConverter](#), [PhotonHandler](#), [PhotonView](#), [PickupItem](#), [PickupItemSimple](#), [PickupItemSyncer](#), [ShowInfoOfPlayer](#), and [SmoothSyncMovement](#).

Properties

- [PhotonView](#) `photonView` [get]
- new [PhotonView](#) `networkView` [get]

This property is only here to notify developers when they use the outdated value.

8.22.1 Detailed Description

This class adds the property `photonView`, while logging a warning when your game still uses the `networkView`.

8.22.2 Property Documentation

8.22.2.1 new PhotonView Photon.MonoBehaviour.networkView [get]

This property is only here to notify developers when they use the outdated value.

If Unity 5.x logs a compiler warning "Use the new keyword if hiding was intended" or "The new keyword is not required", you may suffer from an Editor issue. Try to modify `networkView` with a `if-def` condition:

```
#if UNITY_EDITOR new #endif public PhotonView networkView
```

8.22.2.2 PhotonView Photon.MonoBehaviour.photonView [get]

The documentation for this class was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/Photon↵ Network/[PhotonClasses.cs](#)

8.23 MoveByKeys Class Reference

Very basic component to move a `GameObject` by WASD and Space.

Inherits [Photon.MonoBehaviour](#).

Public Member Functions

- void [Start](#) ()
- bool [IsGrounded](#) ()
- void [FixedUpdate](#) ()

Public Attributes

- float [thrust](#) = 5000
- float [JumpForce](#) = 60000
- float [drag](#) = 0.1f
- float [JumpTimeout](#) = 0.5f
- int [maxPowerups](#) = 5
- int [powerups](#) = 3
- Vector3 [dir](#)
- Vector3 [dirFix](#)
- float [rad](#)
- Time [disableTime](#)
- float [lastPowerup](#) = 300
- Text [powerText](#)

Additional Inherited Members

8.23.1 Detailed Description

Very basic component to move a GameObject by WASD and Space.

Requires a [PhotonView](#). Disables itself on GameObjects that are not owned on Start.

[thrust](#) affects movement-thrust. [JumpForce](#) defines how high the object "jumps". [JumpTimeout](#) defines after how many seconds you can jump again.

8.23.2 Member Function Documentation

8.23.2.1 void [MoveByKeys.FixedUpdate](#) ()

8.23.2.2 bool [MoveByKeys.IsGrounded](#) ()

8.23.2.3 void [MoveByKeys.Start](#) ()

8.23.3 Member Data Documentation

8.23.3.1 Vector3 [MoveByKeys.dir](#)

8.23.3.2 Vector3 [MoveByKeys.dirFix](#)

8.23.3.3 Time [MoveByKeys.disableTime](#)

8.23.3.4 float MoveByKeys.drag = 0.1f

8.23.3.5 float MoveByKeys.JumpForce = 60000

8.23.3.6 float MoveByKeys.JumpTimeout = 0.5f

8.23.3.7 float MoveByKeys.lastPowerup = 300

8.23.3.8 int MoveByKeys.maxPowerups = 5

8.23.3.9 Text MoveByKeys.powerText

8.23.3.10 int MoveByKeys.powerups = 3

8.23.3.11 float MoveByKeys.rad

8.23.3.12 float MoveByKeys.thrust = 5000

The documentation for this class was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/UtilityScripts/Move↵
ByKeys.cs

8.24 OnAwakeUsePhotonView Class Reference

Inherits [Photon.MonoBehaviour](#).

Public Member Functions

- void [OnAwakeRPC](#) ()
- void [OnAwakeRPC](#) (byte myParameter)

Additional Inherited Members

8.24.1 Member Function Documentation

8.24.1.1 void OnAwakeUsePhotonView.OnAwakeRPC ()

8.24.1.2 void OnAwakeUsePhotonView.OnAwakeRPC (byte *myParameter*)

The documentation for this class was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/UtilityScripts/On↵
AwakeUsePhotonView.cs

8.25 OnClickDestroy Class Reference

Implements OnClick to destroy the GameObject it's attached to.

Inherits [Photon.MonoBehaviour](#).

Public Member Functions

- void [OnClick](#) ()
- IEnumerator [DestroyRpc](#) ()

Public Attributes

- bool [DestroyByRpc](#)

Additional Inherited Members

8.25.1 Detailed Description

Implements OnClick to destroy the GameObject it's attached to.

Optionally a RPC is sent to do this.

Using an RPC to Destroy a GameObject allows any player to Destroy a GameObject. But it might cause errors. RPC and the Instantiated GameObject are not fully linked on the server. One might stick in the server without the other.

A buffered RPC gets cleaned up when the sending player leaves the room. This means, the RPC gets lost.

Vice versus, a GameObject Instantiate might get cleaned up when the creating player leaves a room. This way, the GameObject that a RPC targets might become lost.

It makes sense to test those cases. Many are not breaking errors and you just have to be aware of them.

Gets [OnClick\(\)](#) calls by [InputToEvent](#) class attached to a camera.

8.25.2 Member Function Documentation

8.25.2.1 IEnumerator [OnClickDestroy.DestroyRpc](#) ()

8.25.2.2 void [OnClickDestroy.OnClick](#) ()

8.25.3 Member Data Documentation

8.25.3.1 bool [OnClickDestroy.DestroyByRpc](#)

The documentation for this class was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/UtilityScripts/[OnClickDestroy.cs](#)

8.26 OnClickInstantiate Class Reference

Inherits MonoBehaviour.

Public Attributes

- GameObject [Prefab](#)
- int [InstantiateType](#)
- bool [showGui](#)

8.26.1 Member Data Documentation

8.26.1.1 int OnClickInstantiate.InstantiateType

8.26.1.2 GameObject OnClickInstantiate.Prefab

8.26.1.3 bool OnClickInstantiate.showGui

The documentation for this class was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/UtilityScripts/[OnClickInstantiate.cs](#)

8.27 OnClickLoadSomething Class Reference

This component makes it easy to switch scenes or open webpages on click.

Inherits MonoBehaviour.

Public Types

- enum [ResourceTypeOption](#) : byte { [ResourceTypeOption.Scene](#), [ResourceTypeOption.Web](#) }

Public Member Functions

- void [OnClick](#) ()

Public Attributes

- [ResourceTypeOption](#) [ResourceTypeToLoad](#) = [ResourceTypeOption.Scene](#)
- string [ResourceToLoad](#)

8.27.1 Detailed Description

This component makes it easy to switch scenes or open webpages on click.

Requires a [InputToEvent](#) component on the camera to forward clicks on screen.

8.27.2 Member Enumeration Documentation

8.27.2.1 enum `OnClickLoadSomething.ResourceTypeOption` : `byte` `[strong]`

Enumerator

Scene

Web

8.27.3 Member Function Documentation

8.27.3.1 void `OnClickLoadSomething.OnClick` ()

8.27.4 Member Data Documentation

8.27.4.1 string `OnClickLoadSomething.ResourceToLoad`

8.27.4.2 `ResourceTypeOption OnClickLoadSomething.ResourceTypeToLoad = ResourceTypeOption.Scene`

The documentation for this class was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/UtilityScripts/[OnClickLoadSomething.cs](#)

8.28 OnJoinedInstantiate Class Reference

Inherits `MonoBehaviour`.

Public Member Functions

- void [OnJoinedRoom](#) ()

Public Attributes

- Transform [SpawnPosition](#)
- float [PositionOffset](#) = 2.0f
- GameObject[] [PrefabsToInstantiate](#)
- GameObject [PlayerCamera](#)

8.28.1 Member Function Documentation

8.28.1.1 void OnJoinedInstantiate.OnJoinedRoom ()

8.28.2 Member Data Documentation

8.28.2.1 GameObject OnJoinedInstantiate.PlayerCamera

8.28.2.2 float OnJoinedInstantiate.PositionOffset = 2.0f

8.28.2.3 GameObject [] OnJoinedInstantiate.PrefabsToInstantiate

8.28.2.4 Transform OnJoinedInstantiate.SpawnPosition

The documentation for this class was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/UtilityScripts/[OnJoinedInstantiate.cs](#)

8.29 OnStartDelete Class Reference

This component will destroy the GameObject it is attached to (in Start()).

Inherits MonoBehaviour.

8.29.1 Detailed Description

This component will destroy the GameObject it is attached to (in Start()).

The documentation for this class was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/UtilityScripts/[OnStartDelete.cs](#)

8.30 ExitGames.Client.Photon.OperationCode Class Reference

Class for constants.

Public Attributes

- const byte [ExchangeKeysForEncryption](#) = 250
- const byte [Join](#) = 255
(255) Code for OpJoin, to get into a room.
- const byte [Authenticate](#) = 230
(230) Authenticates this peer and connects to a virtual application
- const byte [JoinLobby](#) = 229
(229) Joins lobby (on master)
- const byte [LeaveLobby](#) = 228
(228) Leaves lobby (on master)
- const byte [CreateGame](#) = 227
(227) Creates a game (or fails if name exists)
- const byte [JoinGame](#) = 226
(226) Join game (by name)
- const byte [JoinRandomGame](#) = 225
(225) Joins random game (on master)
- const byte [Leave](#) = (byte)254
(254) Code for OpLeave, to get out of a room.
- const byte [RaiseEvent](#) = (byte)253
(253) Raise event (in a room, for other actors/players)
- const byte [SetProperties](#) = (byte)252
(252) Set Properties (of room or actor/player)
- const byte [GetProperties](#) = (byte)251
(251) Get Properties
- const byte [ChangeGroups](#) = (byte)248
(248) Operation code to change interest groups in Rooms (Lite application and extending ones).
- const byte [FindFriends](#) = 222
(222) Request the rooms and online status for a list of friends (by name, which should be unique).
- const byte [GetLobbyStats](#) = 221
(221) Request statistics about a specific list of lobbies (their user and game count).
- const byte [GetRegions](#) = 220
(220) Get list of regional servers from a NameServer.
- const byte [WebRpc](#) = 219
(219) WebRpc Operation.

8.30.1 Detailed Description

Class for constants.

Contains operation codes. Pun uses these constants internally.

8.30.2 Member Data Documentation

8.30.2.1 const byte ExitGames.Client.Photon.OperationCode.Authenticate = 230

(230) Authenticates this peer and connects to a virtual application

8.30.2.2 `const byte ExitGames.Client.Photon.OperationCode.ChangeGroups = (byte)248`

(248) Operation code to change interest groups in Rooms (Lite application and extending ones).

8.30.2.3 `const byte ExitGames.Client.Photon.OperationCode.CreateGame = 227`

(227) Creates a game (or fails if name exists)

8.30.2.4 `const byte ExitGames.Client.Photon.OperationCode.ExchangeKeysForEncryption = 250`

8.30.2.5 `const byte ExitGames.Client.Photon.OperationCode.FindFriends = 222`

(222) Request the rooms and online status for a list of friends (by name, which should be unique).

8.30.2.6 `const byte ExitGames.Client.Photon.OperationCode.GetLobbyStats = 221`

(221) Request statistics about a specific list of lobbies (their user and game count).

8.30.2.7 `const byte ExitGames.Client.Photon.OperationCode.GetProperties = (byte)251`

(251) Get Properties

8.30.2.8 `const byte ExitGames.Client.Photon.OperationCode.GetRegions = 220`

(220) Get list of regional servers from a NameServer.

8.30.2.9 `const byte ExitGames.Client.Photon.OperationCode.Join = 255`

(255) Code for OpJoin, to get into a room.

8.30.2.10 `const byte ExitGames.Client.Photon.OperationCode.JoinGame = 226`

(226) Join game (by name)

8.30.2.11 `const byte ExitGames.Client.Photon.OperationCode.JoinLobby = 229`

(229) Joins lobby (on master)

8.30.2.12 `const byte ExitGames.Client.Photon.OperationCode.JoinRandomGame = 225`

(225) Joins random game (on master)

8.30.2.13 `const byte ExitGames.Client.Photon.OperationCode.Leave = (byte)254`

(254) Code for OpLeave, to get out of a room.

8.30.2.14 `const byte ExitGames.Client.Photon.OperationCode.LeaveLobby = 228`

(228) Leaves lobby (on master)

8.30.2.15 `const byte ExitGames.Client.Photon.OperationCode.RaiseEvent = (byte)253`

(253) Raise event (in a room, for other actors/players)

8.30.2.16 `const byte ExitGames.Client.Photon.OperationCode.SetProperties = (byte)252`

(252) Set Properties (of room or actor/player)

8.30.2.17 `const byte ExitGames.Client.Photon.OperationCode.WebRpc = 219`

(219) WebRpc Operation.

The documentation for this class was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/Photon↔ Network/[LoadbalancingPeer.cs](#)

8.31 ExitGames.Client.Photon.LoadbalancingPeer.OpJoinRandomRoomParams Class Reference

Public Attributes

- [Hashtable](#) [ExpectedCustomRoomProperties](#)
- [byte](#) [ExpectedMaxPlayers](#)
- [MatchmakingMode](#) [MatchingType](#)
- [TypedLobby](#) [TypedLobby](#)
- [string](#) [SqlLobbyFilter](#)

8.31.1 Member Data Documentation

8.31.1.1 **Hashtable** `ExitGames.Client.Photon.LoadbalancingPeer.OpJoinRandomRoomParams.ExpectedCustomRoom`↔
Properties

8.31.1.2 **byte** `ExitGames.Client.Photon.LoadbalancingPeer.OpJoinRandomRoomParams.ExpectedMaxPlayers`

8.31.1.3 **MatchmakingMode** `ExitGames.Client.Photon.LoadbalancingPeer.OpJoinRandomRoomParams.MatchingType`

8.31.1.4 **string** `ExitGames.Client.Photon.LoadbalancingPeer.OpJoinRandomRoomParams.SqlLobbyFilter`

8.31.1.5 **TypedLobby** `ExitGames.Client.Photon.LoadbalancingPeer.OpJoinRandomRoomParams.TypedLobby`

The documentation for this class was generated from the following file:

- `C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/Photon↔
Network/LoadbalancingPeer.cs`

8.32 ExitGames.Client.Photon.ParameterCode Class Reference

Class for constants.

Public Attributes

- const byte `SuppressRoomEvents` = 237
(237) A bool parameter for creating games. If set to true, no room events are sent to the clients on join and leave. Default: false (and not sent).
- const byte `EmptyRoomTTL` = 236
(236) Time To Live (TTL) for a room when the last player leaves. Keeps room in memory for case a player re-joins soon. In milliseconds.
- const byte `PlayerTTL` = 235
(235) Time To Live (TTL) for an 'actor' in a room. If a client disconnects, this actor is inactive first and removed after this timeout. In milliseconds.
- const byte `EventForward` = 234
(234) Optional parameter of `OpRaiseEvent` to forward the event to some web-service.
- const byte `IsComingBack` = (byte)233
(233) Optional parameter of `OpLeave` in async games. If false, the player does abandons the game (forever). By default players become inactive and can re-join.
- const byte `IsInactive` = (byte)233
(233) Used in `EvLeave` to describe if a user is inactive (and might come back) or not. In async / Turnbased games, inactive is default.
- const byte `CheckUserOnJoin` = (byte)232
(232) Used when creating rooms to define if any userid can join the room only once.
- const byte `ExpectedValues` = (byte)231
(231) Code for "Check And Swap" (CAS) when changing properties.
- const byte `Address` = 230
(230) Address of a (game) server to use.
- const byte `PeerCount` = 229

- (229) Count of players in this application in a rooms (used in stats event)

 - const byte **GameCount** = 228
- (228) Count of games in this application (used in stats event)

 - const byte **MasterPeerCount** = 227
- (227) Count of players on the master server (in this app, looking for rooms)

 - const byte **UserId** = 225
- (225) User's ID

 - const byte **ApplicationId** = 224
- (224) Your application's ID: a name on your own *Photon* or a GUID on the *Photon* Cloud

 - const byte **Position** = 223
- (223) Not used currently (as "Position"). If you get queued before connect, this is your position

 - const byte **MatchMakingType** = 223
- (223) Modifies the matchmaking algorithm used for OpJoinRandom. Allowed parameter values are defined in enum MatchmakingMode.

 - const byte **GameList** = 222
- (222) List of RoomInfos about open / listed rooms

 - const byte **Secret** = 221
- (221) Internally used to establish encryption

 - const byte **AppVersion** = 220
- (220) Version of your application

 - const byte **AzureNodeInfo** = 210
- (210) Internally used in case of hosting by Azure

 - const byte **AzureLocalNodeId** = 209
- (209) Internally used in case of hosting by Azure

 - const byte **AzureMasterNodeId** = 208
- (208) Internally used in case of hosting by Azure

 - const byte **RoomName** = (byte)255
- (255) Code for the gameId/roomName (a unique name per room). Used in OpJoin and similar.

 - const byte **Broadcast** = (byte)250
- (250) Code for broadcast parameter of OpSetProperties method.

 - const byte **ActorList** = (byte)252
- (252) Code for list of players in a room. Currently not used.

 - const byte **ActorNr** = (byte)254
- (254) Code of the Actor of an operation. Used for property get and set.

 - const byte **PlayerProperties** = (byte)249
- (249) Code for property set (Hashtable).

 - const byte **CustomEventContent** = (byte)245
- (245) Code of data/custom content of an event. Used in OpRaiseEvent.

 - const byte **Data** = (byte)245
- (245) Code of data of an event. Used in OpRaiseEvent.

 - const byte **Code** = (byte)244
- (244) Code used when sending some code-related parameter, like OpRaiseEvent's event-code.

 - const byte **GameProperties** = (byte)248
- (248) Code for property set (Hashtable).

 - const byte **Properties** = (byte)251
- (251) Code for property-set (Hashtable).

 - const byte **TargetActorNr** = (byte)253
- (253) Code of the target Actor of an operation. Used for property set. Is 0 for game

 - const byte **ReceiverGroup** = (byte)246
- (246) Code to select the receivers of events (used in Lite, Operation RaiseEvent).

 - const byte **Cache** = (byte)247

- (247) Code for caching events while raising them.

 - const byte [CleanupCacheOnLeave](#) = (byte)241

(241) Bool parameter of CreateGame Operation. If true, server cleans up roomcache of leaving players (their cached events get removed).
- const byte [Group](#) = 240

(240) Code for "group" operation-parameter (as used in Op RaiseEvent).
- const byte [Remove](#) = 239

(239) The "Remove" operation-parameter can be used to remove something from a list. E.g. remove groups from player's interest groups.
- const byte [PublishUserId](#) = 239

(239) Used in Op Join to define if UserIds of the players are broadcast in the room. Useful for FindFriends and reserving slots for expected users.
- const byte [Add](#) = 238

(238) The "Add" operation-parameter can be used to add something to some list or set. E.g. add groups to player's interest groups.
- const byte [Info](#) = 218

(218) Content for [EventCode.ErrorInfo](#) and internal debug operations.
- const byte [ClientAuthenticationType](#) = 217

(217) This key's (byte) value defines the target custom authentication type/service the client connects with. Used in OpAuthenticate
- const byte [ClientAuthenticationParams](#) = 216

(216) This key's (string) value provides parameters sent to the custom authentication type/service the client connects with. Used in OpAuthenticate
- const byte [JoinMode](#) = 215

(215) Makes the server create a room if it doesn't exist. OpJoin uses this to always enter a room, unless it exists and is full/closed.
- const byte [ClientAuthenticationData](#) = 214

(214) This key's (string or byte[]) value provides parameters sent to the custom authentication service setup in [Photon Dashboard](#). Used in OpAuthenticate
- const byte [MasterClientId](#) = (byte)203

(203) Code for MasterClientId, which is synced by server.
- const byte [FindFriendsRequestList](#) = (byte)1

(1) Used in Op FindFriends request. Value must be string[] of friends to look up.
- const byte [FindFriendsResponseOnlineList](#) = (byte)1

(1) Used in Op FindFriends response. Contains bool[] list of online states (false if not online).
- const byte [FindFriendsResponseRoomIdList](#) = (byte)2

(2) Used in Op FindFriends response. Contains string[] of room names ("" where not known or no room joined).
- const byte [LobbyName](#) = (byte)213

(213) Used in matchmaking-related methods and when creating a room to name a lobby (to join or to attach a room to).
- const byte [LobbyType](#) = (byte)212

(212) Used in matchmaking-related methods and when creating a room to define the type of a lobby. Combined with the lobby name this identifies the lobby.
- const byte [LobbyStats](#) = (byte)211

(211) This (optional) parameter can be sent in Op Authenticate to turn on Lobby Stats (info about lobby names and their user- and game-counts). See: [PhotonNetwork.Lobbies](#)
- const byte [Region](#) = (byte)210

(210) Used for region values in OpAuth and OpGetRegions.
- const byte [UriPath](#) = 209

(209) Path of the WebRPC that got called. Also known as "WebRpc Name". Type: string.
- const byte [WebRpcParameters](#) = 208

(208) Parameters for a WebRPC as: Dictionary<string, object>. This will get serialized to JSON.
- const byte [WebRpcReturnCode](#) = 207

- (207) *ReturnCode* for the WebRPC, as sent by the web service (not by [Photon](#), which uses [ErrorCode](#)). Type: byte.

 - const byte [WebRpcReturnMessage](#) = 206

(206) *Message* returned by WebRPC server. Analog to [Photon](#)'s debug message. Type: string.

 - const byte [CacheSliceIndex](#) = 205

(205) *Used to define a "slice" for cached events. Slices can easily be removed from cache. Type: int.*

 - const byte [Plugins](#) = 204

Notifies the server of the expected plugin setup.

 - const byte [PluginName](#) = 201

(201) *Notifies user about name of plugin load to game*

 - const byte [PluginVersion](#) = 200

(200) *Notifies user about version of plugin load to game*

8.32.1 Detailed Description

Class for constants.

Codes for parameters of Operations and Events. Pun uses these constants internally.

8.32.2 Member Data Documentation

8.32.2.1 const byte ExitGames.Client.Photon.ParameterCode.ActorList = (byte)252

(252) Code for list of players in a room. Currently not used.

8.32.2.2 const byte ExitGames.Client.Photon.ParameterCode.ActorNr = (byte)254

(254) Code of the Actor of an operation. Used for property get and set.

8.32.2.3 const byte ExitGames.Client.Photon.ParameterCode.Add = 238

(238) The "Add" operation-parameter can be used to add something to some list or set. E.g. add groups to player's interest groups.

8.32.2.4 const byte ExitGames.Client.Photon.ParameterCode.Address = 230

(230) Address of a (game) server to use.

8.32.2.5 const byte ExitGames.Client.Photon.ParameterCode.ApplicationId = 224

(224) Your application's ID: a name on your own [Photon](#) or a GUID on the [Photon](#) Cloud

8.32.2.6 const byte ExitGames.Client.Photon.ParameterCode.AppVersion = 220

(220) Version of your application

8.32.2.7 `const byte ExitGames.Client.Photon.ParameterCode.AzureLocalNodeId = 209`

(209) Internally used in case of hosting by Azure

8.32.2.8 `const byte ExitGames.Client.Photon.ParameterCode.AzureMasterNodeId = 208`

(208) Internally used in case of hosting by Azure

8.32.2.9 `const byte ExitGames.Client.Photon.ParameterCode.AzureNodeInfo = 210`

(210) Internally used in case of hosting by Azure

8.32.2.10 `const byte ExitGames.Client.Photon.ParameterCode.Broadcast = (byte)250`

(250) Code for broadcast parameter of `OpSetProperties` method.

8.32.2.11 `const byte ExitGames.Client.Photon.ParameterCode.Cache = (byte)247`

(247) Code for caching events while raising them.

8.32.2.12 `const byte ExitGames.Client.Photon.ParameterCode.CacheSliceIndex = 205`

(205) Used to define a "slice" for cached events. Slices can easily be removed from cache. Type: `int`.

8.32.2.13 `const byte ExitGames.Client.Photon.ParameterCode.CheckUserOnJoin = (byte)232`

(232) Used when creating rooms to define if any `userid` can join the room only once.

8.32.2.14 `const byte ExitGames.Client.Photon.ParameterCode.CleanupCacheOnLeave = (byte)241`

(241) `Bool` parameter of `CreateGame` Operation. If true, server cleans up roomcache of leaving players (their cached events get removed).

8.32.2.15 `const byte ExitGames.Client.Photon.ParameterCode.ClientAuthenticationData = 214`

(214) This key's (string or `byte[]`) value provides parameters sent to the custom authentication service setup in [Photon](#) Dashboard. Used in `OpAuthenticate`

8.32.2.16 `const byte ExitGames.Client.Photon.ParameterCode.ClientAuthenticationParams = 216`

(216) This key's (string) value provides parameters sent to the custom authentication type/service the client connects with. Used in `OpAuthenticate`

8.32.2.17 `const byte ExitGames.Client.Photon.ParameterCode.ClientAuthenticationType = 217`

(217) This key's (byte) value defines the target custom authentication type/service the client connects with. Used in `OpAuthenticate`

8.32.2.18 `const byte ExitGames.Client.Photon.ParameterCode.Code = (byte)244`

(244) Code used when sending some code-related parameter, like `OpRaiseEvent`'s event-code.

This is not the same as the Operation's code, which is no longer sent as part of the parameter Dictionary in [Photon 3](#).

8.32.2.19 `const byte ExitGames.Client.Photon.ParameterCode.CustomEventContent = (byte)245`

(245) Code of data/custom content of an event. Used in `OpRaiseEvent`.

8.32.2.20 `const byte ExitGames.Client.Photon.ParameterCode.Data = (byte)245`

(245) Code of data of an event. Used in `OpRaiseEvent`.

8.32.2.21 `const byte ExitGames.Client.Photon.ParameterCode.EmptyRoomTTL = 236`

(236) Time To Live (TTL) for a room when the last player leaves. Keeps room in memory for case a player re-joins soon. In milliseconds.

8.32.2.22 `const byte ExitGames.Client.Photon.ParameterCode.EventForward = 234`

(234) Optional parameter of `OpRaiseEvent` to forward the event to some web-service.

8.32.2.23 `const byte ExitGames.Client.Photon.ParameterCode.ExpectedValues = (byte)231`

(231) Code for "Check And Swap" (CAS) when changing properties.

8.32.2.24 `const byte ExitGames.Client.Photon.ParameterCode.FindFriendsRequestList = (byte)1`

(1) Used in `Op FindFriends` request. Value must be `string[]` of friends to look up.

8.32.2.25 `const byte ExitGames.Client.Photon.ParameterCode.FindFriendsResponseOnlineList = (byte)1`

(1) Used in `Op FindFriends` response. Contains `bool[]` list of online states (false if not online).

8.32.2.26 `const byte ExitGames.Client.Photon.ParameterCode.FindFriendsResponseRoomIdList = (byte)2`

(2) Used in Op FindFriends response. Contains string[] of room names ("" where not known or no room joined).

8.32.2.27 `const byte ExitGames.Client.Photon.ParameterCode.GameCount = 228`

(228) Count of games in this application (used in stats event)

8.32.2.28 `const byte ExitGames.Client.Photon.ParameterCode.GameList = 222`

(222) List of RoomInfos about open / listed rooms

8.32.2.29 `const byte ExitGames.Client.Photon.ParameterCode.GameProperties = (byte)248`

(248) Code for property set (Hashtable).

8.32.2.30 `const byte ExitGames.Client.Photon.ParameterCode.Group = 240`

(240) Code for "group" operation-parameter (as used in Op RaiseEvent).

8.32.2.31 `const byte ExitGames.Client.Photon.ParameterCode.Info = 218`

(218) Content for [EventCode.ErrorInfo](#) and internal debug operations.

8.32.2.32 `const byte ExitGames.Client.Photon.ParameterCode.IsComingBack = (byte)233`

(233) Optional parameter of OpLeave in async games. If false, the player does abandons the game (forever). By default players become inactive and can re-join.

8.32.2.33 `const byte ExitGames.Client.Photon.ParameterCode.IsInactive = (byte)233`

(233) Used in EvLeave to describe if a user is inactive (and might come back) or not. In async / Turnbased games, inactive is default.

8.32.2.34 `const byte ExitGames.Client.Photon.ParameterCode.JoinMode = 215`

(215) Makes the server create a room if it doesn't exist. OpJoin uses this to always enter a room, unless it exists and is full/closed.

(215) The JoinMode enum defines which variant of joining a room will be executed: Join only if available, create if not exists or re-join.

Replaces CreateIfNotExists which was only a bool-value.

8.32.2.35 `const byte ExitGames.Client.Photon.ParameterCode.LobbyName = (byte)213`

(213) Used in matchmaking-related methods and when creating a room to name a lobby (to join or to attach a room to).

8.32.2.36 `const byte ExitGames.Client.Photon.ParameterCode.LobbyStats = (byte)211`

(211) This (optional) parameter can be sent in Op Authenticate to turn on Lobby Stats (info about lobby names and their user- and game-counts). See: [PhotonNetwork.Lobbies](#)

8.32.2.37 `const byte ExitGames.Client.Photon.ParameterCode.LobbyType = (byte)212`

(212) Used in matchmaking-related methods and when creating a room to define the type of a lobby. Combined with the lobby name this identifies the lobby.

8.32.2.38 `const byte ExitGames.Client.Photon.ParameterCode.MasterClientId = (byte)203`

(203) Code for MasterClientId, which is synced by server.

When sent as op-parameter this is code 203.

Tightly related to [GamePropertyKey.MasterClientId](#).

8.32.2.39 `const byte ExitGames.Client.Photon.ParameterCode.MasterPeerCount = 227`

(227) Count of players on the master server (in this app, looking for rooms)

8.32.2.40 `const byte ExitGames.Client.Photon.ParameterCode.MatchMakingType = 223`

(223) Modifies the matchmaking algorithm used for OpJoinRandom. Allowed parameter values are defined in enum MatchmakingMode.

8.32.2.41 `const byte ExitGames.Client.Photon.ParameterCode.PeerCount = 229`

(229) Count of players in this application in a rooms (used in stats event)

8.32.2.42 `const byte ExitGames.Client.Photon.ParameterCode.PlayerProperties = (byte)249`

(249) Code for property set (Hashtable).

8.32.2.43 `const byte ExitGames.Client.Photon.ParameterCode.PlayerTTL = 235`

(235) Time To Live (TTL) for an 'actor' in a room. If a client disconnects, this actor is inactive first and removed after this timeout. In milliseconds.

8.32.2.44 `const byte ExitGames.Client.Photon.ParameterCode.PluginName = 201`

(201) Informs user about name of plugin load to game

8.32.2.45 `const byte ExitGames.Client.Photon.ParameterCode.Plugins = 204`

Informs the server of the expected plugin setup.

The operation will fail in case of a plugin mismatch returning error code `PluginMismatch 32751 (0x7FFF - 16)`. Setting `string[]{}` means the client expects no plugin to be setup. Note: for backwards compatibility null omits any check.

8.32.2.46 `const byte ExitGames.Client.Photon.ParameterCode.PluginVersion = 200`

(200) Informs user about version of plugin load to game

8.32.2.47 `const byte ExitGames.Client.Photon.ParameterCode.Position = 223`

(223) Not used currently (as "Position"). If you get queued before connect, this is your position

8.32.2.48 `const byte ExitGames.Client.Photon.ParameterCode.Properties = (byte)251`

(251) Code for property-set (Hashtable).

This key is used when sending only one set of properties. If either [ActorProperties](#) or `GameProperties` are used (or both), check those keys.

8.32.2.49 `const byte ExitGames.Client.Photon.ParameterCode.PublishUserId = 239`

(239) Used in Op Join to define if UserIds of the players are broadcast in the room. Useful for FindFriends and reserving slots for expected users.

8.32.2.50 `const byte ExitGames.Client.Photon.ParameterCode.ReceiverGroup = (byte)246`

(246) Code to select the receivers of events (used in Lite, Operation RaiseEvent).

8.32.2.51 `const byte ExitGames.Client.Photon.ParameterCode.Region = (byte)210`

(210) Used for region values in OpAuth and OpGetRegions.

8.32.2.52 `const byte ExitGames.Client.Photon.ParameterCode.Remove = 239`

(239) The "Remove" operation-parameter can be used to remove something from a list. E.g. remove groups from player's interest groups.

8.32.2.53 `const byte ExitGames.Client.Photon.ParameterCode.RoomName = (byte)255`

(255) Code for the gameId/roomName (a unique name per room). Used in OpJoin and similar.

8.32.2.54 `const byte ExitGames.Client.Photon.ParameterCode.Secret = 221`

(221) Internally used to establish encryption

8.32.2.55 `const byte ExitGames.Client.Photon.ParameterCode.SuppressRoomEvents = 237`

(237) A bool parameter for creating games. If set to true, no room events are sent to the clients on join and leave. Default: false (and not sent).

8.32.2.56 `const byte ExitGames.Client.Photon.ParameterCode.TargetActorNr = (byte)253`

(253) Code of the target Actor of an operation. Used for property set. Is 0 for game

8.32.2.57 `const byte ExitGames.Client.Photon.ParameterCode.UriPath = 209`

(209) Path of the WebRPC that got called. Also known as "WebRpc Name". Type: string.

8.32.2.58 `const byte ExitGames.Client.Photon.ParameterCode.UserId = 225`

(225) User's ID

8.32.2.59 `const byte ExitGames.Client.Photon.ParameterCode.WebRpcParameters = 208`

(208) Parameters for a WebRPC as: Dictionary<string, object>. This will get serialized to JSON.

8.32.2.60 `const byte ExitGames.Client.Photon.ParameterCode.WebRpcReturnCode = 207`

(207) ReturnCode for the WebRPC, as sent by the web service (not by [Photon](#), which uses [ErrorCode](#)). Type: byte.

8.32.2.61 `const byte ExitGames.Client.Photon.ParameterCode.WebRpcReturnMessage = 206`

(206) Message returned by WebRPC server. Analog to [Photon](#)'s debug message. Type: string.

The documentation for this class was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/Photon↔ Network/[LoadbalancingPeer.cs](#)

8.33 PhotonAnimatorView Class Reference

This class helps you to synchronize Mecanim animations Simply add the component to your GameObject and make sure that the [PhotonAnimatorView](#) is added to the list of observed components

Inherits MonoBehaviour.

Classes

- class [SynchronizedLayer](#)
- class [SynchronizedParameter](#)

Public Types

- enum [ParameterType](#) { [ParameterType.Float](#) = 1, [ParameterType.Int](#) = 3, [ParameterType.Bool](#) = 4, [ParameterType.Trigger](#) = 9 }
- enum [SynchronizeType](#) { [SynchronizeType.Disabled](#) = 0, [SynchronizeType.Discrete](#) = 1, [SynchronizeType.Continuous](#) = 2 }

Public Member Functions

- bool [DoesLayerSynchronizeTypeExist](#) (int layerIndex)
Check if a specific layer is configured to be synchronize
- bool [DoesParameterSynchronizeTypeExist](#) (string name)
Check if the specified parameter is configured to be synchronized
- List< [SynchronizedLayer](#) > [GetSynchronizedLayers](#) ()
Get a list of all synchronized layers
- List< [SynchronizedParameter](#) > [GetSynchronizedParameters](#) ()
Get a list of all synchronized parameters
- [SynchronizeType](#) [GetLayerSynchronizeType](#) (int layerIndex)
Gets the type how the layer is synchronized
- [SynchronizeType](#) [GetParameterSynchronizeType](#) (string name)
Gets the type how the parameter is synchronized
- void [SetLayerSynchronized](#) (int layerIndex, [SynchronizeType](#) synchronizeType)
Sets the how a layer should be synchronized
- void [SetParameterSynchronized](#) (string name, [ParameterType](#) type, [SynchronizeType](#) synchronizeType)
Sets the how a parameter should be synchronized

8.33.1 Detailed Description

This class helps you to synchronize Mecanim animations Simply add the component to your GameObject and make sure that the [PhotonAnimatorView](#) is added to the list of observed components

8.33.2 Member Enumeration Documentation

8.33.2.1 enum PhotonAnimatorView.ParameterType [strong]

Enumerator

Float
Int
Bool
Trigger

8.33.2.2 enum PhotonAnimatorView.SynchronizeType [strong]

Enumerator

Disabled
Discrete
Continuous

8.33.3 Member Function Documentation

8.33.3.1 bool PhotonAnimatorView.DoesLayerSynchronizeTypeExist (int *layerIndex*)

Check if a specific layer is configured to be synchronize

Parameters

<i>layerIndex</i>	Index of the layer.
-------------------	---------------------

Returns

True if the layer is synchronized

8.33.3.2 bool PhotonAnimatorView.DoesParameterSynchronizeTypeExist (string *name*)

Check if the specified parameter is configured to be synchronized

Parameters

<i>name</i>	The name of the parameter.
-------------	----------------------------

Returns

True if the parameter is synchronized

8.33.3.3 **SynchronizeType** PhotonAnimatorView.GetLayerSynchronizeType (int *layerIndex*)

Gets the type how the layer is synchronized

Parameters

<i>layerIndex</i>	Index of the layer.
-------------------	---------------------

Returns

Disabled/Discrete/Continuous

8.33.3.4 **SynchronizeType** PhotonAnimatorView.GetParameterSynchronizeType (string *name*)

Gets the type how the parameter is synchronized

Parameters

<i>name</i>	The name of the parameter.
-------------	----------------------------

Returns

Disabled/Discrete/Continuous

8.33.3.5 **List<SynchronizedLayer>** PhotonAnimatorView.GetSynchronizedLayers ()

Get a list of all synchronized layers

Returns

List of [SynchronizedLayer](#) objects

8.33.3.6 **List<SynchronizedParameter>** PhotonAnimatorView.GetSynchronizedParameters ()

Get a list of all synchronized parameters

Returns

List of [SynchronizedParameter](#) objects

8.33.3.7 **void** PhotonAnimatorView.SetLayerSynchronized (int *layerIndex*, **SynchronizeType** *synchronizeType*)

Sets the how a layer should be synchronized

Parameters

<i>layerIndex</i>	Index of the layer.
<i>synchronizeType</i>	Disabled/Discrete/Continuous

8.33.3.8 void PhotonAnimatorView.SetParameterSynchronized (string *name*, ParameterType *type*, SynchronizeType *synchronizeType*)

Sets the how a parameter should be synchronized

Parameters

<i>name</i>	The name of the parameter.
<i>type</i>	The type of the parameter.
<i>synchronizeType</i>	Disabled/Discrete/Continuous

The documentation for this class was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/Photon↔
Network/Views/[PhotonAnimatorView.cs](#)

8.34 PhotonAnimatorViewEditor Class Reference

Inherits Editor.

Public Member Functions

- override void [OnInspectorGUI](#) ()

8.34.1 Member Function Documentation

8.34.1.1 override void PhotonAnimatorViewEditor.OnInspectorGUI ()

The documentation for this class was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Editor/Photon↔
Network/Views/[PhotonAnimatorViewEditor.cs](#)

8.35 PhotonConverter Class Reference

Inherits [Photon.MonoBehaviour](#).

Static Public Member Functions

- static void [RunConversion](#) ()
- static void [PickFolderAndConvertScripts](#) ()
- static List< string > [GetScriptsInFolder](#) (string folder)
- static void [ConvertRpcAttribute](#) (string path)

default path: "Assets"

Additional Inherited Members

8.35.1 Member Function Documentation

8.35.1.1 static void PhotonConverter.ConvertRpcAttribute (string *path*) [static]

default path: "Assets"

8.35.1.2 static List<string> PhotonConverter.GetScriptsInFolder (string *folder*) [static]

8.35.1.3 static void PhotonConverter.PickFolderAndConvertScripts () [static]

8.35.1.4 static void PhotonConverter.RunConversion () [static]

The documentation for this class was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Editor/Photon↔ Network/[PhotonConverter.cs](#)

8.36 PhotonEditor Class Reference

Inherits EditorWindow.

Public Member Functions

- [PhotonEditor](#) ()

Static Public Member Functions

- static void [UpdateRpcList](#) ()
- static void [ClearRpcList](#) ()
- static System.Type[] [GetAllSubTypesInScripts](#) (System.Type aBaseClass)

Static Public Attributes

- static [PunWizardText](#) CurrentLang = new [PunWizardText](#)()

Protected Member Functions

- void [Update](#) ()
- virtual void [OnGUI](#) ()
- virtual void [UiSetupApp](#) ()
- virtual void [UiMainWizard](#) ()
- virtual void [RegisterWithEmail](#) (string email)

Static Protected Member Functions

- static void [MenuItemOpenWizard](#) ()
- static void [MenuItemHighlightSettings](#) ()
- static void [ShowRegistrationWizard](#) ()

Creates an Editor window, showing the cloud-registration wizard for [Photon](#) (entry point to setup PUN).

Protected Attributes

- Vector2 [scrollPos](#) = Vector2.zero

Static Protected Attributes

- static Type [WindowType](#) = typeof ([PhotonEditor](#))
- static [AccountService.Origin](#) [RegisterOrigin](#) = [AccountService.Origin.Pun](#)
- static string [DocumentationLocation](#) = "Assets/Photon Unity Networking/[PhotonNetwork](#)-Documentation.pdf"
- static string [UrlFreeLicense](#) = "https://www.photonengine.com/en/OnPremise/Dashboard"
- static string [UrlDevNet](#) = "http://doc.photonengine.com/en/pun/current"
- static string [UrlForum](#) = "http://forum.exitgames.com"
- static string [UrlCompare](#) = "http://doc.photonengine.com/en/realtime/current/getting-started/onpremise-or-saas"
- static string [UrlHowToSetup](#) = "http://doc.photonengine.com/en/onpremise/current/getting-started/photon-server-in-5min"
- static string [UrlAppIDExplained](#) = "http://doc.photonengine.com/en/realtime/current/getting-started/obtain-your-app-id"
- static string [UrlAccountPage](#) = "https://www.photonengine.com/Account/SignIn?email="
- static string [UrlCloudDashboard](#) = "https://www.photonengine.com/Dashboard?email="

8.36.1 Constructor & Destructor Documentation

8.36.1.1 PhotonEditor.PhotonEditor ()

8.36.2 Member Function Documentation

8.36.2.1 static void PhotonEditor.ClearRpcList () [static]

8.36.2.2 static System.Type [] PhotonEditor.GetAllSubTypesInScripts (System.Type aBaseClass) [static]

8.36.2.3 static void PhotonEditor.MenuItemHighlightSettings () [static], [protected]

8.36.2.4 static void PhotonEditor.MenuItemOpenWizard () [static], [protected]

8.36.2.5 virtual void PhotonEditor.OnGUI () [protected], [virtual]

8.36.2.6 virtual void PhotonEditor.RegisterWithEmail (string email) [protected], [virtual]

8.36.2.7 static void PhotonEditor.ShowRegistrationWizard () [static], [protected]

Creates an Editor window, showing the cloud-registration wizard for [Photon](#) (entry point to setup PUN).

8.36.2.8 `virtual void PhotonEditor.UiMainWizard ()` [protected],[virtual]

8.36.2.9 `virtual void PhotonEditor.UiSetupApp ()` [protected],[virtual]

8.36.2.10 `void PhotonEditor.Update ()` [protected]

8.36.2.11 `static void PhotonEditor.UpdateRpcList ()` [static]

8.36.3 Member Data Documentation

8.36.3.1 `PunWizardText PhotonEditor.CurrentLang = new PunWizardText()` [static]

8.36.3.2 `string PhotonEditor.DocumentationLocation = "Assets/Photon Unity Networking/PhotonNetwork-Documentation.pdf"` [static],[protected]

8.36.3.3 `AccountService.Origin PhotonEditor.RegisterOrigin = AccountService.Origin.Pun` [static],[protected]

8.36.3.4 `Vector2 PhotonEditor.scrollPos = Vector2.zero` [protected]

8.36.3.5 `string PhotonEditor.UrlAccountPage = "https://www.photonengine.com/Account/SignIn?email="` [static],[protected]

8.36.3.6 `string PhotonEditor.UrlAppIDExplained = "http://doc.photonengine.com/en/realtime/current/getting-started/obtain-your-app-id"` [static],[protected]

8.36.3.7 `string PhotonEditor.UrlCloudDashboard = "https://www.photonengine.com/Dashboard?email="` [static],[protected]

8.36.3.8 `string PhotonEditor.UrlCompare = "http://doc.photonengine.com/en/realtime/current/getting-started/onpremise-or-saas"` [static],[protected]

8.36.3.9 `string PhotonEditor.UrlDevNet = "http://doc.photonengine.com/en/pun/current"` [static],[protected]

8.36.3.10 `string PhotonEditor.UrlForum = "http://forum.exitgames.com"` [static],[protected]

8.36.3.11 `string PhotonEditor.UrlFreeLicense = "https://www.photonengine.com/en/OnPremise/Dashboard"` [static],[protected]

8.36.3.12 `string PhotonEditor.UrlHowToSetup = "http://doc.photonengine.com/en/onpremise/current/getting-started/photon-server-in-5min"` [static],[protected]

8.36.3.13 `Type PhotonEditor.WindowType = typeof (PhotonEditor)` [static],[protected]

The documentation for this class was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Editor/PhotonNetwork/PhotonEditor.cs

8.37 PhotonGUI Class Reference

Static Public Member Functions

- static void [ContainerHeader](#) (string headline)
- static bool [ContainerHeaderToggle](#) (string headline, bool toggle)
- static bool [ContainerHeaderFoldout](#) (string headline, bool foldout)
- static Rect [ContainerBody](#) (float height)
- static bool [AddButton](#) ()
- static void [DrawSplitter](#) (Rect position)
- static void [DrawGizmoOptions](#) (Rect position, string label, SerializedProperty gizmoEnabledProperty, SerializedProperty gizmoColorProperty, SerializedProperty gizmoTypeProperty, SerializedProperty gizmoSizeProperty)

Properties

- static GUIStyle [DefaultTitleStyle](#) [get]
- static GUIStyle [DefaultContainerStyle](#) [get]
- static GUIStyle [DefaultAddButtonStyle](#) [get]
- static GUIStyle [DefaultRemoveButtonStyle](#) [get]
- static GUIStyle [DefaultContainerRowStyle](#) [get]
- static GUIStyle [FoldoutBold](#) [get]
- static GUIStyle [RichLabel](#) [get]
- static Texture2D [HelpIcon](#) [get]

8.37.1 Member Function Documentation

8.37.1.1 static bool PhotonGUI.AddButton () [static]

8.37.1.2 static Rect PhotonGUI.ContainerBody (float *height*) [static]

8.37.1.3 static void PhotonGUI.ContainerHeader (string *headline*) [static]

8.37.1.4 static bool PhotonGUI.ContainerHeaderFoldout (string *headline*, bool *foldout*) [static]

8.37.1.5 static bool PhotonGUI.ContainerHeaderToggle (string *headline*, bool *toggle*) [static]

8.37.1.6 static void PhotonGUI.DrawGizmoOptions (Rect *position*, string *label*, SerializedProperty *gizmoEnabledProperty*, SerializedProperty *gizmoColorProperty*, SerializedProperty *gizmoTypeProperty*, SerializedProperty *gizmoSizeProperty*) [static]

8.37.1.7 static void PhotonGUI.DrawSplitter (Rect *position*) [static]

8.37.2 Property Documentation

8.37.2.1 GUIStyle PhotonGUI.DefaultAddButtonStyle [static], [get]

8.37.2.2 GUIStyle PhotonGUI.DefaultContainerRowStyle [static], [get]

8.37.2.3 GUIStyle PhotonGUI.DefaultContainerStyle [static], [get]

8.37.2.4 GUIStyle PhotonGUI.DefaultRemoveButtonStyle [static], [get]

8.37.2.5 GUIStyle PhotonGUI.DefaultTitleStyle [static], [get]

8.37.2.6 GUIStyle PhotonGUI.FoldoutBold [static], [get]

8.37.2.7 Texture2D PhotonGUI.HelpIcon [static], [get]

8.37.2.8 GUIStyle PhotonGUI.RichLabel [static], [get]

The documentation for this class was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Editor/Photon↔ Network/[PhotonGUI.cs](#)

8.38 PhotonLagSimulationGui Class Reference

This MonoBehaviour is a basic GUI for the [Photon](#) client's network-simulation feature.

Inherits MonoBehaviour.

Public Member Functions

- void [Start](#) ()
- void [OnGUI](#) ()

Public Attributes

- Rect [WindowRect](#) = new Rect(0, 100, 120, 100)
Positioning rect for window.
- int [WindowId](#) = 101
Unity GUI Window ID (must be unique or will cause issues).
- bool [Visible](#) = true
Shows or hides GUI (does not affect settings).

Properties

- PhotonPeer [Peer](#) [get, set]
The peer currently in use (to set the network simulation).

8.38.1 Detailed Description

This MonoBehaviour is a basic GUI for the [Photon](#) client's network-simulation feature.

It can modify lag (fixed delay), jitter (random lag) and packet loss.

8.38.2 Member Function Documentation

8.38.2.1 void PhotonLagSimulationGui.OnGUI ()

8.38.2.2 void PhotonLagSimulationGui.Start ()

8.38.3 Member Data Documentation

8.38.3.1 bool PhotonLagSimulationGui.Visible = true

Shows or hides GUI (does not affect settings).

8.38.3.2 int PhotonLagSimulationGui.WindowId = 101

Unity GUI Window ID (must be unique or will cause issues).

8.38.3.3 Rect PhotonLagSimulationGui.WindowRect = new Rect(0, 100, 120, 100)

Positioning rect for window.

8.38.4 Property Documentation

8.38.4.1 PhotonPeer PhotonLagSimulationGui.Peer [get], [set]

The peer currently in use (to set the network simulation).

The documentation for this class was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/Photon↔ Network/[PhotonLagSimulationGui.cs](#)

8.39 PhotonMessageInfo Class Reference

Container class for info about a particular message, RPC or update.

Public Member Functions

- [PhotonMessageInfo](#) ()
Initializes a new instance of the [PhotonMessageInfo](#) class.
- [PhotonMessageInfo](#) ([PhotonPlayer](#) player, int timestamp, [PhotonView](#) view)
- override string [ToString](#) ()

Public Attributes

- [PhotonPlayer](#) sender
- [PhotonView](#) photonView

Properties

- double [timestamp](#) [get]

8.39.1 Detailed Description

Container class for info about a particular message, RPC or update.

8.39.2 Constructor & Destructor Documentation

8.39.2.1 [PhotonMessageInfo.PhotonMessageInfo](#) ()

Initializes a new instance of the [PhotonMessageInfo](#) class.

To create an empty messageinfo only!

8.39.2.2 [PhotonMessageInfo.PhotonMessageInfo](#) ([PhotonPlayer](#) player, int timestamp, [PhotonView](#) view)

8.39.3 Member Function Documentation

8.39.3.1 override string [PhotonMessageInfo.ToString](#) ()

8.39.4 Member Data Documentation

8.39.4.1 [PhotonView](#) [PhotonMessageInfo.photonView](#)

8.39.4.2 [PhotonPlayer](#) [PhotonMessageInfo.sender](#)

8.39.5 Property Documentation

8.39.5.1 double [PhotonMessageInfo.timestamp](#) [get]

The documentation for this class was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/Photon↔ Network/[PhotonClasses.cs](#)

8.40 PhotonNetwork Class Reference

The main class to use the [PhotonNetwork](#) plugin.

Public Member Functions

- delegate void [EventCallback](#) (byte eventCode, object content, int senderId)

Defines the delegate usable in OnEventCall.

Static Public Member Functions

- static void [SwitchToProtocol](#) (ConnectionProtocol cp)
While offline, the network protocol can be switched (which affects the ports you can use to connect).
- static bool [ConnectUsingSettings](#) (string gameVersion)
Connect to [Photon](#) as configured in the editor (saved in PhotonServerSettings file).
- static bool [ConnectToMaster](#) (string masterServerAddress, int port, string appId, string gameVersion)
Connect to a [Photon](#) Master Server by address, port, appId and game(client) version.
- static bool [ConnectToBestCloudServer](#) (string gameVersion)
Connect to the [Photon](#) Cloud region with the lowest ping (on platforms that support Unity's Ping).
- static bool [ConnectToRegion](#) (CloudRegionCode region, string gameVersion)
Connects to the [Photon](#) Cloud region of choice.
- static void [OverrideBestCloudServer](#) (CloudRegionCode region)
Overwrites the region that is used for [ConnectToBestCloudServer\(string gameVersion\)](#).
- static void [RefreshCloudServerRating](#) ()
Pings all cloud servers again to find the one with best ping (currently).
- static void [NetworkStatisticsReset](#) ()
Resets the traffic stats and re-enables them.
- static string [NetworkStatisticsToString](#) ()
Only available when NetworkStatisticsEnabled was used to gather some stats.
- static void [InitializeSecurity](#) ()
Used for compatibility with Unity networking only.
- static void [Disconnect](#) ()
Makes this client disconnect from the photon server, a process that leaves any room and calls OnDisconnected←FromPhoton on completion.
- static bool [FindFriends](#) (string[] friendsToFind)
Requests the rooms and online status for a list of friends and saves the result in [PhotonNetwork.Friends](#).
- static bool [CreateRoom](#) (string roomName)
Creates a room with given name but fails if this room(name) is existing already.
- static bool [CreateRoom](#) (string roomName, RoomOptions roomOptions, TypedLobby typedLobby)
Creates a room but fails if this room is existing already.
- static bool [JoinRoom](#) (string roomName)
Join room by roomname and on success calls OnJoinedRoom().
- static bool [JoinOrCreateRoom](#) (string roomName, RoomOptions roomOptions, TypedLobby typedLobby)
Lets you either join a named room or create it on the fly - you don't have to know if someone created the room already.
- static bool [JoinRandomRoom](#) ()
Joins any available room of the currently used lobby and fails if none is available.
- static bool [JoinRandomRoom](#) (Hashtable expectedCustomRoomProperties, byte expectedMaxPlayers)
Attempts to join an open room with fitting, custom properties but fails if none is currently available.

- static bool [JoinRandomRoom](#) ([Hashtable](#) expectedCustomRoomProperties, byte expectedMaxPlayers, [MatchingMode](#) matchingType, [TypedLobby](#) typedLobby, string sqlLobbyFilter)
Attempts to join an open room with fitting, custom properties but fails if none is currently available.
- static bool [JoinLobby](#) ()
On MasterServer this joins the default lobby which list rooms currently in use.
- static bool [JoinLobby](#) ([TypedLobby](#) typedLobby)
On a Master Server you can join a lobby to get lists of available rooms.
- static bool [LeaveLobby](#) ()
Leave a lobby to stop getting updates about available rooms.
- static bool [LeaveRoom](#) ()
Leave the current room and return to the Master Server where you can join or create rooms (see remarks).
- static [RoomInfo](#)[] [GetRoomList](#) ()
Gets currently known rooms as [RoomInfo](#) array.
- static void [SetPlayerCustomProperties](#) ([Hashtable](#) customProperties)
Sets this (local) player's properties and synchronizes them to the other players (don't modify them directly).
- static void [RemovePlayerCustomProperties](#) (string[] customPropertiesToDelete)
Locally removes Custom Properties of "this" player.
- static bool [RaiseEvent](#) (byte eventCode, object eventContent, bool sendReliable, [RaiseEventOptions](#) options)
Sends fully customizable events in a room.
- static int [AllocateViewID](#) ()
Allocates a viewID that's valid for the current/local player.
- static int [AllocateSceneViewID](#) ()
Enables the Master Client to allocate a viewID that is valid for scene objects.
- static void [UnAllocateViewID](#) (int viewID)
Unregister a viewID (of manually instantiated and destroyed networked objects).
- static [GameObject](#) [Instantiate](#) (string prefabName, [Vector3](#) position, [Quaternion](#) rotation, int group)
Instantiate a prefab over the network.
- static [GameObject](#) [Instantiate](#) (string prefabName, [Vector3](#) position, [Quaternion](#) rotation, int group, object[] data)
Instantiate a prefab over the network.
- static [GameObject](#) [InstantiateSceneObject](#) (string prefabName, [Vector3](#) position, [Quaternion](#) rotation, int group, object[] data)
Instantiate a scene-owned prefab over the network.
- static int [GetPing](#) ()
The current roundtrip time to the photon server.
- static void [FetchServerTimestamp](#) ()
Refreshes the server timestamp (async operation, takes a roundtrip).
- static void [SendOutgoingCommands](#) ()
Can be used to immediately send the RPCs and Instantiates just called, so they are on their way to the other players.
- static bool [CloseConnection](#) ([PhotonPlayer](#) kickPlayer)
Request a client to disconnect (KICK).
- static bool [SetMasterClient](#) ([PhotonPlayer](#) masterClientPlayer)
Asks the server to assign another player as Master Client of your current room.
- static void [Destroy](#) ([PhotonView](#) targetView)
Network-Destroy the GameObject associated with the [PhotonView](#), unless the [PhotonView](#) is static or not under this client's control.
- static void [Destroy](#) ([GameObject](#) targetGo)
Network-Destroy the GameObject, unless it is static or not under this client's control.
- static void [DestroyPlayerObjects](#) ([PhotonPlayer](#) targetPlayer)
Network-Destroy all GameObjects, PhotonViews and their RPCs of targetPlayer.
- static void [DestroyPlayerObjects](#) (int targetPlayerId)

- Network-Destroy all GameObjects, PhotonViews and their RPCs of this player (by ID).*

 - static void [DestroyAll](#) ()
- Network-Destroy all GameObjects, PhotonViews and their RPCs in the room.*

 - static void [RemoveRPCs](#) ([PhotonPlayer](#) targetPlayer)

Remove all buffered RPCs from server that were sent by targetPlayer.
- static void [RemoveRPCs](#) ([PhotonView](#) targetPhotonView)

Remove all buffered RPCs from server that were sent via targetPhotonView.
- static void [RemoveRPCsInGroup](#) (int targetGroup)

Remove all buffered RPCs from server that were sent in the targetGroup, if this is the Master Client or if this controls the individual [PhotonView](#).
- static void [CacheSendMonoMessageTargets](#) (Type type)

Populates SendMonoMessageTargets with currently existing GameObjects that have a Component of type.
- static HashSet< [GameObject](#) > [FindGameObjectsWithComponent](#) (Type type)

Finds the GameObjects with Components of a specific type (using FindObjectsOfType).
- static void [SetReceivingEnabled](#) (int group, bool enabled)

Enable/disable receiving on given group (applied to PhotonViews)
- static void [SetReceivingEnabled](#) (int[] enableGroups, int[] disableGroups)

Enable/disable receiving on given groups (applied to PhotonViews)
- static void [SetSendingEnabled](#) (int group, bool enabled)

Enable/disable sending on given group (applied to PhotonViews)
- static void [SetSendingEnabled](#) (int[] enableGroups, int[] disableGroups)

Enable/disable sending on given groups (applied to PhotonViews)
- static void [SetLevelPrefix](#) (short prefix)

Sets level prefix for PhotonViews instantiated later on.
- static void [LoadLevel](#) (int levelNumber)

Wraps loading a level to pause the network message-queue.
- static void [LoadLevel](#) (string levelName)

Wraps loading a level to pause the network message-queue.
- static bool [WebRpc](#) (string name, object parameters)

This operation makes [Photon](#) call your custom web-service by name (path) with the given parameters.

Public Attributes

- const string [versionPUN](#) = "1.65"

Version number of PUN. Also used in GameVersion to separate client version from each other.

Static Public Attributes

- static readonly int [MAX_VIEW_IDS](#) = 1000

The maximum number of assigned PhotonViews per player (or scene).
- static [ServerSettings](#) [PhotonServerSettings](#) = ([ServerSettings](#))Resources.Load(PhotonNetwork.server↵ SettingsAssetFile, typeof([ServerSettings](#)))

Serialized server settings, written by the Setup Wizard for use in ConnectUsingSettings.
- static bool [InstantiateInRoomOnly](#) = true

If true, Instantiate methods will check if you are in a room and fail if you are not.
- static [PhotonLogLevel](#) [logLevel](#) = [PhotonLogLevel.ErrorsOnly](#)

Network log level.
- static float [precisionForVectorSynchronization](#) = 0.000099f

The minimum difference that a Vector2 or Vector3(e.g.
- static float [precisionForQuaternionSynchronization](#) = 1.0f

The minimum angle that a rotation needs to change before we send it via a [PhotonView](#)'s `OnSerialize/Observe` Component.

- static float [precisionForFloatSynchronization](#) = 0.01f

The minimum difference between floats before we send it via a [PhotonView](#)'s `OnSerialize/Observe` Component.

- static bool [UseRpcMonoBehaviourCache](#)

While enabled, the `MonoBehaviours` on which we call RPCs are cached, avoiding costly `GetComponent<MonoBehaviour>()` calls.

- static bool [UsePrefabCache](#) = true

While enabled (true), `Instantiate` uses [PhotonNetwork.PrefabCache](#) to keep game objects in memory (improving instantiation of the same prefab).

- static Dictionary< string, GameObject > [PrefabCache](#) = new Dictionary<string, GameObject>()

Keeps references to `GameObjects` for frequent instantiation (out of memory instead of loading the Resources).

- static HashSet< GameObject > [SendMonoMessageTargets](#)

If not null, this is the (exclusive) list of `GameObjects` that get called by `PUN SendMonoMessage()`.

- static Type [SendMonoMessageTargetType](#) = typeof(MonoBehaviour)

Defines which classes can contain `PUN Callback` implementations.

- static int [maxConnections](#)

Only used in Unity Networking. In `PUN`, set the number of players in [PhotonNetwork.CreateRoom](#).

- static float [BackgroundTimeout](#) = 0.0f

Defines after how many seconds `PUN` will close a connection, after Unity's `OnApplicationPause(true)` call.

- static [EventCallback OnEventCall](#)

Register your `RaiseEvent` handling methods here by using "+=".

Properties

- static string [gameVersion](#) [get, set]

Version string for your this build.

- static string [ServerAddress](#) [get]

Currently used server address (no matter if master or game server).

- static bool [connected](#) [get]

False until you connected to [Photon](#) initially.

- static bool [connecting](#) [get]

True when you called `ConnectUsingSettings` (or similar) until the low level connection to [Photon](#) gets established.

- static bool [connectedAndReady](#) [get]

A refined version of `connected` which is true only if your connection to the server is ready to accept operations like join, leave, etc.

- static [ConnectionState connectionState](#) [get]

Simplified connection state

- static [PeerState connectionStateDetailed](#) [get]

Detailed connection state (ignorant of `PUN`, so it can be "disconnected" while switching servers).

- static [ServerConnection Server](#) [get]

The server (type) this client is currently connected or connecting to.

- static [AuthenticationValues AuthValues](#) [get, set]

A user's authentication values used during connect for Custom Authentication with [Photon](#) (and a custom service/community).

- static [Room room](#) [get]

Get the room we're currently in.

- static [PhotonPlayer player](#) [get]

The local [PhotonPlayer](#).

- static [PhotonPlayer masterClient](#) [get]

The Master Client of the current room or null (outside of rooms).

- static string [playerName](#) [get, set]
Set to synchronize the player's nickname with everyone in the room(s) you enter.
- static [PhotonPlayer\[\]](#) [playerList](#) [get]
The list of players in the current room, including the local player.
- static [PhotonPlayer\[\]](#) [otherPlayers](#) [get]
The list of players in the current room, excluding the local player.
- static List< [FriendInfo](#) > [Friends](#) [get, set]
Read-only list of friends, their online status and the room they are in.
- static int [FriendsListAge](#) [get]
Age of friend list info (in milliseconds).
- static [IPunPrefabPool](#) [PrefabPool](#) [get, set]
An Object Pool can be used to keep and reuse instantiated object instances.
- static bool [offlineMode](#) [get, set]
Offline mode can be set to re-use your multiplayer code in singleplayer game modes.
- static bool [automaticallySyncScene](#) [get, set]
Defines if all clients in a room should load the same level as the Master Client (if that used [PhotonNetwork.LoadLevel](#)).
- static bool [autoCleanUpPlayerObjects](#) [get, set]
This setting defines per room, if network-instantiated GameObjects (with [PhotonView](#)) get cleaned up when the creator of it leaves.
- static bool [autoJoinLobby](#) [get, set]
Set in PhotonServerSettings asset.
- static bool [EnableLobbyStatistics](#) [get, set]
Set in PhotonServerSettings asset.
- static List< [TypedLobbyInfo](#) > [LobbyStatistics](#) [get]
If turned on, the Master Server will provide information about active lobbies for this application.
- static bool [insideLobby](#) [get]
True while this client is in a lobby.
- static [TypedLobby](#) [lobby](#) [get, set]
The lobby that will be used when PUN joins a lobby or creates a game.
- static int [sendRate](#) [get, set]
Defines how many times per second [PhotonNetwork](#) should send a package.
- static int [sendRateOnSerialize](#) [get, set]
Defines how many times per second OnPhotonSerialize should be called on PhotonViews.
- static bool [isMessageQueueRunning](#) [get, set]
Can be used to pause dispatching of incoming events (RPCs, Instantiates and anything else incoming).
- static int [unreliableCommandsLimit](#) [get, set]
Used once per dispatch to limit unreliable commands per channel (so after a pause, many channels can still cause a lot of unreliable commands)
- static double [time](#) [get]
[Photon](#) network time, synched with the server.
- static int [ServerTimestamp](#) [get]
The current server's millisecond timestamp.
- static bool [isMasterClient](#) [get]
Are we the master client?
- static bool [inRoom](#) [get]
Is true while being in a room (connectionStateDetailed == [PeerState.Joined](#)).
- static bool [isNonMasterClientInRoom](#) [get]
True if we are in a room (client) and NOT the room's masterclient
- static int [countOfPlayersOnMaster](#) [get]
The count of players currently looking for a room (available on MasterServer in 5sec intervals).
- static int [countOfPlayersInRooms](#) [get]

- Count of users currently playing your app in some room (sent every 5sec by Master Server).*
- static int [countOfPlayers](#) [get]
- The count of players currently using this application (available on MasterServer in 5sec intervals).*
- static int [countOfRooms](#) [get]
- The count of rooms currently in use (available on MasterServer in 5sec intervals).*
- static bool [NetworkStatisticsEnabled](#) [get, set]
- Enables or disables the collection of statistics about this client's traffic.*
- static int [ResentReliableCommands](#) [get]
- Count of commands that got repeated (due to local repeat-timing before an ACK was received).*
- static bool [CrcCheckEnabled](#) [get, set]
- Crc checks can be useful to detect and avoid issues with broken datagrams. Can be enabled while not connected.*
- static int [PacketLossByCrcCheck](#) [get]
- If CrcCheckEnabled, this counts the incoming packages that don't have a valid CRC checksum and got rejected.*
- static int [MaxResendsBeforeDisconnect](#) [get, set]
- Defines the number of times a reliable message can be resent before not getting an ACK for it will trigger a disconnect.*
- static int [QuickResends](#) [get, set]
- In case of network loss, reliable messages can be repeated quickly up to 3 times.*

8.40.1 Detailed Description

The main class to use the [PhotonNetwork](#) plugin.

This class is static.

8.40.2 Member Function Documentation

8.40.2.1 static int PhotonNetwork.AllocateSceneViewID () [static]

Enables the Master Client to allocate a viewID that is valid for scene objects.

Returns

A viewID that can be used for a new [PhotonView](#) or -1 in case of an error.

8.40.2.2 static int PhotonNetwork.AllocateViewID () [static]

Allocates a viewID that's valid for the current/local player.

Returns

A viewID that can be used for a new [PhotonView](#).

8.40.2.3 static void PhotonNetwork.CacheSendMonoMessageTargets (Type type) [static]

Populates SendMonoMessageTargets with currently existing GameObjects that have a Component of type.

Parameters

<i>type</i>	If null, this will use <code>SendMessageTargets</code> as component-type (MonoBehaviour by default).
-------------	--

8.40.2.4 static bool PhotonNetwork.CloseConnection (PhotonPlayer kickPlayer) [static]

Request a client to disconnect (KICK).

Only the master client can do this

Only the target player gets this event. That player will disconnect automatically, which is what the others will notice, too.

Parameters

<i>kickPlayer</i>	The PhotonPlayer to kick.
-------------------	---

8.40.2.5 static bool PhotonNetwork.ConnectToBestCloudServer (string gameVersion) [static]

Connect to the [Photon](#) Cloud region with the lowest ping (on platforms that support Unity's Ping).

Will save the result of pinging all cloud servers in PlayerPrefs. Calling this the first time can take +-2 seconds. The ping result can be overridden via [PhotonNetwork.OverrideBestCloudServer\(..\)](#) This call can take up to 2 seconds if it is the first time you are using this, all cloud servers will be pinged to check for the best region.

The PUN Setup Wizard stores your appId in a settings file and applies a server address/port. To connect to the [Photon](#) Cloud, a valid AppId must be in the settings file (shown in the [Photon](#) Cloud Dashboard). <https://www.exitgames.com/dashboard>

Connecting to the [Photon](#) Cloud might fail due to:

- Invalid AppId (calls: [OnFailedToConnectToPhoton\(\)](#). check exact AppId value)
- Network issues (calls: [OnFailedToConnectToPhoton\(\)](#))
- Invalid region (calls: [OnConnectionFail\(\)](#) with [DisconnectCause.InvalidRegion](#))
- Subscription CCU limit reached (calls: [OnConnectionFail\(\)](#) with [DisconnectCause.MaxCcuReached](#). also calls: [OnPhotonMaxCcuReached\(\)](#))

More about the connection limitations: <http://doc.exitgames.com/en/pun>

Parameters

<i>gameVersion</i>	This client's version number. Users are separated from each other by gameversion (which allows you to make breaking changes).
--------------------	---

Returns

If this client is going to connect to cloud server based on ping. Even if true, this does not guarantee a connection but the attempt is being made.

8.40.2.6 `static bool PhotonNetwork.ConnectToMaster (string masterServerAddress, int port, string appId, string gameVersion) [static]`

Connect to a [Photon](#) Master Server by address, port, appId and game(client) version.

To connect to the [Photon](#) Cloud, a valid AppId must be in the settings file (shown in the [Photon](#) Cloud Dashboard). <https://www.exitgames.com/dashboard>

Connecting to the [Photon](#) Cloud might fail due to:

- Invalid AppId (calls: [OnFailedToConnectToPhoton\(\)](#). check exact AppId value)
- Network issues (calls: [OnFailedToConnectToPhoton\(\)](#))
- Invalid region (calls: [OnConnectionFail\(\)](#) with [DisconnectCause.InvalidRegion](#))
- Subscription CCU limit reached (calls: [OnConnectionFail\(\)](#) with [DisconnectCause.MaxCcuReached](#). also calls: [OnPhotonMaxCcuReached\(\)](#))

More about the connection limitations: <http://doc.exitgames.com/en/pun>

Parameters

<i>masterServerAddress</i>	The server's address (either your own or Photon Cloud address).
<i>port</i>	The server's port to connect to.
<i>appId</i>	Your application ID (Photon Cloud provides you with a GUID for your game).
<i>gameVersion</i>	This client's version number. Users are separated by gameversion (which allows you to make breaking changes).

8.40.2.7 `static bool PhotonNetwork.ConnectToRegion (CloudRegionCode region, string gameVersion) [static]`

Connects to the [Photon](#) Cloud region of choice.

8.40.2.8 `static bool PhotonNetwork.ConnectUsingSettings (string gameVersion) [static]`

Connect to [Photon](#) as configured in the editor (saved in PhotonServerSettings file).

This method will disable offlineMode (which won't destroy any instantiated GOs) and it will set isMessageQueueRunning to true.

Your server configuration is created by the PUN Wizard and contains the AppId and region for [Photon](#) Cloud games and the server address if you host [Photon](#) yourself. These settings usually don't change often.

To ignore the config file and connect anywhere call: [PhotonNetwork.ConnectToMaster](#).

To connect to the [Photon](#) Cloud, a valid AppId must be in the settings file (shown in the [Photon](#) Cloud Dashboard). <https://www.exitgames.com/dashboard>

Connecting to the [Photon](#) Cloud might fail due to:

- Invalid AppId (calls: [OnFailedToConnectToPhoton\(\)](#). check exact AppId value)
- Network issues (calls: [OnFailedToConnectToPhoton\(\)](#))
- Invalid region (calls: [OnConnectionFail\(\)](#) with [DisconnectCause.InvalidRegion](#))
- Subscription CCU limit reached (calls: [OnConnectionFail\(\)](#) with [DisconnectCause.MaxCcuReached](#). also calls: [OnPhotonMaxCcuReached\(\)](#))

More about the connection limitations: <http://doc.exitgames.com/en/pun>

Parameters

<i>gameVersion</i>	This client's version number. Users are separated from each other by gameversion (which allows you to make breaking changes).
--------------------	---

8.40.2.9 static bool PhotonNetwork.CreateRoom (string *roomName*) [static]

Creates a room with given name but fails if this room(name) is existing already.

Creates random name for roomName null.

If you don't want to create a unique room-name, pass null or "" as name and the server will assign a roomName (a GUID as string).

The created room is automatically placed in the currently used lobby (if any) or the default-lobby if you didn't explicitly join one.

Call this only on the master server. Internally, the master will respond with a server-address (and roomName, if needed). Both are used internally to switch to the assigned game server and roomName.

[PhotonNetwork.autoCleanUpPlayerObjects](#) will become this room's AutoCleanUp property and that's used by all clients that join this room.

Parameters

<i>roomName</i>	Unique name of the room to create.
-----------------	------------------------------------

8.40.2.10 static bool PhotonNetwork.CreateRoom (string *roomName*, RoomOptions *roomOptions*, TypedLobby *typedLobby*) [static]

Creates a room but fails if this room is existing already.

Can only be called on Master Server.

When successful, this calls the callbacks OnCreatedRoom and OnJoinedRoom (the latter, cause you join as first player). If the room can't be created (because it exists already), OnPhotonCreateRoomFailed gets called.

If you don't want to create a unique room-name, pass null or "" as name and the server will assign a roomName (a GUID as string).

Rooms can be created in any number of lobbies. Those don't have to exist before you create a room in them (they get auto-created on demand). Lobbies can be useful to split room lists on the server-side already. That can help

keep the room lists short and manageable. If you set a `typedLobby` parameter, the room will be created in that lobby (no matter if you are active in any). If you don't set a `typedLobby`, the room is automatically placed in the currently active lobby (if any) or the default-lobby.

Call this only on the master server. Internally, the master will respond with a server-address (and `roomName`, if needed). Both are used internally to switch to the assigned game server and `roomName`.

[PhotonNetwork.autoCleanUpPlayerObjects](#) will become this room's `autoCleanUp` property and that's used by all clients that join this room.

Parameters

<i>roomName</i>	Unique name of the room to create. Pass null or "" to make the server generate a name.
<i>roomOptions</i>	Common options for the room like <code>maxPlayers</code> , initial custom room properties and similar. See RoomOptions type..
<i>typedLobby</i>	If null, the room is automatically created in the currently used lobby (which is "default" when you didn't join one explicitly).

8.40.2.11 static void PhotonNetwork.Destroy (PhotonView targetView) [static]

Network-Destroy the `GameObject` associated with the [PhotonView](#), unless the [PhotonView](#) is static or not under this client's control.

Destroying a networked `GameObject` while in a [Room](#) includes:

- Removal of the `Instantiate` call from the server's room buffer.
- Removing RPCs buffered for `PhotonViews` that got created indirectly with the [PhotonNetwork.Instantiate](#) call.
- Sending a message to other clients to remove the `GameObject` also (affected by network lag).

Usually, when you leave a room, the GOs get destroyed automatically. If you have to destroy a GO while not in a room, the `Destroy` is only done locally.

Destroying networked objects works only if they got created with [PhotonNetwork.Instantiate\(\)](#). Objects loaded with a scene are ignored, no matter if they have [PhotonView](#) components.

The `GameObject` must be under this client's control:

- Instantiated and owned by this client.
- Instantiated objects of players who left the room are controlled by the Master Client.
- Scene-owned game objects are controlled by the Master Client.
- `GameObject` can be destroyed while client is not in a room.

Returns

Nothing. Check error debug log for any issues.

8.40.2.12 static void PhotonNetwork.Destroy (GameObject *targetGo*) [static]

Network-Destroy the GameObject, unless it is static or not under this client's control.

Destroying a networked GameObject includes:

- Removal of the Instantiate call from the server's room buffer.
- Removing RPCs buffered for PhotonViews that got created indirectly with the [PhotonNetwork.Instantiate](#) call.
- Sending a message to other clients to remove the GameObject also (affected by network lag).

Usually, when you leave a room, the GOs get destroyed automatically. If you have to destroy a GO while not in a room, the Destroy is only done locally.

Destroying networked objects works only if they got created with [PhotonNetwork.Instantiate\(\)](#). Objects loaded with a scene are ignored, no matter if they have [PhotonView](#) components.

The GameObject must be under this client's control:

- Instantiated and owned by this client.
- Instantiated objects of players who left the room are controlled by the Master Client.
- Scene-owned game objects are controlled by the Master Client.
- GameObject can be destroyed while client is not in a room.

Returns

Nothing. Check error debug log for any issues.

8.40.2.13 static void PhotonNetwork.DestroyAll () [static]

Network-Destroy all GameObjects, PhotonViews and their RPCs in the room.

Removes anything buffered from the server. Can only be called by Master Client (for anyone).

Can only be called by Master Client (for anyone). Unlike the Destroy methods, this will remove anything from the server's room buffer. If your game buffers anything beyond Instantiate and RPC calls, that will be cleaned as well from server.

Destroying all includes:

- Remove anything from the server's room buffer (Instantiate, RPCs, anything buffered).
- Sending a message to other clients to destroy everything locally, too (affected by network lag).

Destroying networked objects works only if they got created with [PhotonNetwork.Instantiate\(\)](#). Objects loaded with a scene are ignored, no matter if they have [PhotonView](#) components.

Returns

Nothing. Check error debug log for any issues.

8.40.2.14 static void PhotonNetwork.DestroyPlayerObjects (PhotonPlayer targetPlayer) [static]

Network-Destroy all GameObjects, PhotonViews and their RPCs of targetPlayer.

Can only be called on local player (for "self") or Master Client (for anyone).

Destroying a networked GameObject includes:

- Removal of the Instantiate call from the server's room buffer.
- Removing RPCs buffered for PhotonViews that got created indirectly with the [PhotonNetwork.Instantiate](#) call.
- Sending a message to other clients to remove the GameObject also (affected by network lag).

Destroying networked objects works only if they got created with [PhotonNetwork.Instantiate\(\)](#). Objects loaded with a scene are ignored, no matter if they have [PhotonView](#) components.

Returns

Nothing. Check error debug log for any issues.

8.40.2.15 static void PhotonNetwork.DestroyPlayerObjects (int targetPlayerId) [static]

Network-Destroy all GameObjects, PhotonViews and their RPCs of this player (by ID).

Can only be called on local player (for "self") or Master Client (for anyone).

Destroying a networked GameObject includes:

- Removal of the Instantiate call from the server's room buffer.
- Removing RPCs buffered for PhotonViews that got created indirectly with the [PhotonNetwork.Instantiate](#) call.
- Sending a message to other clients to remove the GameObject also (affected by network lag).

Destroying networked objects works only if they got created with [PhotonNetwork.Instantiate\(\)](#). Objects loaded with a scene are ignored, no matter if they have [PhotonView](#) components.

Returns

Nothing. Check error debug log for any issues.

8.40.2.16 static void PhotonNetwork.Disconnect () [static]

Makes this client disconnect from the photon server, a process that leaves any room and calls [OnDisconnectedFromPhoton](#) on completion.

When you disconnect, the client will send a "disconnecting" message to the server. This speeds up leave/disconnect messages for players in the same room as you (otherwise the server would timeout this client's connection). When used in offlineMode, the state-change and event-call [OnDisconnectedFromPhoton](#) are immediate. Offline mode is set to false as well. Once disconnected, the client can connect again. Use [ConnectUsingSettings](#).

8.40.2.17 delegate void PhotonNetwork.EventCallback (byte eventCode, object content, int senderId)

Defines the delegate usable in [OnEventCall](#).

Any eventCode < 200 will be forwarded to your delegate(s).

Parameters

<i>eventCode</i>	The code assignend to the incoming event.
<i>content</i>	The content the sender put into the event.
<i>senderId</i>	The ID of the player who sent the event. It might be 0, if the "room" sent the event.

8.40.2.18 `static void PhotonNetwork.FetchServerTimestamp () [static]`

Refreshes the server timestamp (async operation, takes a roundtrip).

Can be useful if a bad connection made the timestamp unusable or imprecise.

8.40.2.19 `static bool PhotonNetwork.FindFriends (string[] friendsToFind) [static]`

Requests the rooms and online status for a list of friends and saves the result in [PhotonNetwork.Friends](#).

Works only on Master Server to find the rooms played by a selected list of users.

The result will be stored in [PhotonNetwork.Friends](#) when available. That list is initialized on first use of [OpFindFriends](#) (before that, it is null). To refresh the list, call FindFriends again (in 5 seconds or 10 or 20).

Users identify themselves by setting a unique username via [PhotonNetwork.playerName](#) or by [PhotonNetwork.AuthValues](#). The user id set in AuthValues overrides the playerName, so make sure you know the ID your friends use to authenticate. The AuthValues are sent in [OpAuthenticate](#) when you connect, so the AuthValues must be set before you connect!

Note: Changing a player's name doesn't make sense when using a friend list.

The list of friends must be fetched from some other source (not provided by [Photon](#)).

Internal: The server response includes 2 arrays of info (each index matching a friend from the request): [ParameterCode.FindFriendsResponseOnlineList](#) = bool[] of online states [ParameterCode.FindFriendsResponseRoomIdList](#) = string[] of room names (empty string if not in a room)

Parameters

<i>friendsToFind</i>	Array of friend (make sure to use unique playerName or AuthValues).
----------------------	---

Returns

If the operation could be sent (requires connection, only one request is allowed at any time). Always false in offline mode.

8.40.2.20 `static HashSet<GameObject> PhotonNetwork.FindGameObjectsWithComponent (Type type) [static]`

Finds the GameObjects with Components of a specific type (using [FindObjectOfType](#)).

Parameters

<i>type</i>	Type must be a Component
-------------	--------------------------

Returns

HashSet with GameObjects that have a specific type of Component.

8.40.2.21 `static int PhotonNetwork.GetPing () [static]`

The current roundtrip time to the photon server.

Returns

Roundtrip time (to server and back).

8.40.2.22 `static RoomInfo [] PhotonNetwork.GetRoomList () [static]`

Gets currently known rooms as [RoomInfo](#) array.

This is available and updated while in a lobby (check `insideLobby`).

This list is a cached copy of the internal rooms list so it can be accessed each frame if needed. Per [RoomInfo](#) you can check if the room is full by comparing `playerCount` and `maxPlayers` before you allow a join.

The name of a room must be used to join it (via `JoinRoom`).

Closed rooms are also listed by lobbies but they can't be joined. While in a room, any player can set [Room.visible](#) and [Room.open](#) to hide rooms from matchmaking and close them.

Returns

[RoomInfo](#)[] of current rooms in lobby.

8.40.2.23 `static void PhotonNetwork.InitializeSecurity () [static]`

Used for compatibility with Unity networking only.

Encryption is automatically initialized while connecting.

8.40.2.24 `static GameObject PhotonNetwork.Instantiate (string prefabName, Vector3 position, Quaternion rotation, int group) [static]`

Instantiate a prefab over the network.

This prefab needs to be located in the root of a "Resources" folder.

Instead of using prefabs in the Resources folder, you can manually `Instantiate` and assign `PhotonViews`. See doc.

Parameters

<i>prefabName</i>	Name of the prefab to instantiate.
<i>position</i>	Position Vector3 to apply on instantiation.
<i>rotation</i>	Rotation Quaternion to apply on instantiation.
<i>group</i>	The group for this PhotonView .

Returns

The new instance of a GameObject with initialized [PhotonView](#).

8.40.2.25 `static GameObject PhotonNetwork.Instantiate (string prefabName, Vector3 position, Quaternion rotation, int group, object[] data) [static]`

Instantiate a prefab over the network.

This prefab needs to be located in the root of a "Resources" folder.

Instead of using prefabs in the Resources folder, you can manually Instantiate and assign PhotonViews. See doc.

Parameters

<i>prefabName</i>	Name of the prefab to instantiate.
<i>position</i>	Position Vector3 to apply on instantiation.
<i>rotation</i>	Rotation Quaternion to apply on instantiation.
<i>group</i>	The group for this PhotonView .
<i>data</i>	Optional instantiation data. This will be saved to it's PhotonView.instantiationData .

Returns

The new instance of a GameObject with initialized [PhotonView](#).

8.40.2.26 `static GameObject PhotonNetwork.InstantiateSceneObject (string prefabName, Vector3 position, Quaternion rotation, int group, object[] data) [static]`

Instantiate a scene-owned prefab over the network.

The PhotonViews will be controllable by the MasterClient. This prefab needs to be located in the root of a "↔Resources" folder.

Only the master client can Instantiate scene objects. Instead of using prefabs in the Resources folder, you can manually Instantiate and assign PhotonViews. See doc.

Parameters

<i>prefabName</i>	Name of the prefab to instantiate.
<i>position</i>	Position Vector3 to apply on instantiation.
<i>rotation</i>	Rotation Quaternion to apply on instantiation.
<i>group</i>	The group for this PhotonView .
<i>data</i>	Optional instantiation data. This will be saved to it's PhotonView.instantiationData .

Returns

The new instance of a `GameObject` with initialized `PhotonView`.

8.40.2.27 static bool PhotonNetwork.JoinLobby () [static]

On MasterServer this joins the default lobby which list rooms currently in use.

The room list is sent and refreshed by the server. You can access this cached list by `PhotonNetwork.GetRoomList()`.

Per room you should check if it's full or not before joining. `Photon` also lists rooms that are full, unless you close and hide them (`room.open = false` and `room.visible = false`).

In best case, you make your clients join random games, as described here: <http://doc.exitgames.com/en/realtime/current/reference/matchmaking-and-lobby>

You can show your current players and room count without joining a lobby (but you must be on the master server). Use: `countOfPlayers`, `countOfPlayersOnMaster`, `countOfPlayersInRooms` and `countOfRooms`.

You can use more than one lobby to keep the room lists shorter. See `JoinLobby(TypedLobby lobby)`. When creating new rooms, they will be "attached" to the currently used lobby or the default lobby.

You can use `JoinRandomRoom` without being in a lobby! Set `autoJoinLobby = false` before you connect, to not join a lobby. In that case, the connect-workflow will call `OnConnectedToMaster` (if you implement it) when it's done.

8.40.2.28 static bool PhotonNetwork.JoinLobby (TypedLobby typedLobby) [static]

On a Master Server you can join a lobby to get lists of available rooms.

The room list is sent and refreshed by the server. You can access this cached list by `PhotonNetwork.GetRoomList()`.

Any client can "make up" any lobby on the fly. Splitting rooms into multiple lobbies will keep each list shorter. However, having too many lists might ruin the matchmaking experience.

In best case, you create a limited number of lobbies. For example, create a lobby per game-mode: "koth" for king of the hill and "ffa" for free for all, etc.

There is no listing of lobbies at the moment.

Sql-typed lobbies offer a different filtering model for random matchmaking. This might be more suited for skillbased-games. However, you will also need to follow the conventions for naming filterable properties in sql-lobbies! Both is explained in the matchmaking doc linked below.

In best case, you make your clients join random games, as described here: <http://confluence.exitgames.com/display/PTN/Op+JoinRandomGame>

Per room you should check if it's full or not before joining. `Photon` does list rooms that are full, unless you close and hide them (`room.open = false` and `room.visible = false`).

You can show your games current players and room count without joining a lobby (but you must be on the master server). Use: `countOfPlayers`, `countOfPlayersOnMaster`, `countOfPlayersInRooms` and `countOfRooms`.

When creating new rooms, they will be "attached" to the currently used lobby or the default lobby.

You can use `JoinRandomRoom` without being in a lobby! Set `autoJoinLobby = false` before you connect, to not join a lobby. In that case, the connect-workflow will call `OnConnectedToMaster` (if you implement it) when it's done.

Parameters

<i>typedLobby</i>	A typed lobby to join (must have name and type).
-------------------	--

8.40.2.29 `static bool PhotonNetwork.JoinOrCreateRoom (string roomName, RoomOptions roomOptions, TypedLobby typedLobby) [static]`

Lets you either join a named room or create it on the fly - you don't have to know if someone created the room already.

This makes it easier for groups of players to get into the same room. Once the group exchanged a *roomName*, any player can call `JoinOrCreateRoom` and it doesn't matter who actually joins or creates the room.

The parameters *roomOptions* and *typedLobby* are only used when the room actually gets created by this client. You know if this client created a room, if you get a callback `OnCreatedRoom` (before `OnJoinedRoom` gets called as well).

Parameters

<i>roomName</i>	Name of the room to join. Must be non null.
<i>roomOptions</i>	Options for the room, in case it does not exist yet. Else these values are ignored.
<i>typedLobby</i>	Lobby you want a new room to be listed in. Ignored if the room was existing and got joined.

Returns

If the operation got queued and will be sent.

8.40.2.30 `static bool PhotonNetwork.JoinRandomRoom () [static]`

Joins any available room of the currently used lobby and fails if none is available.

Rooms can be created in arbitrary lobbies which get created on demand. You can join rooms from any lobby without actually joining the lobby. Use the `JoinRandomRoom` overload with [TypedLobby](#) parameter.

This method will only match rooms attached to one lobby! If you use many lobbies, you might have to repeat `JoinRandomRoom`, to find some fitting room. This method looks up a room in the currently active lobby or (if no lobby is joined) in the default lobby.

If this fails, you can still create a room (and make this available for the next who uses `JoinRandomRoom`). Alternatively, try again in a moment.

8.40.2.31 `static bool PhotonNetwork.JoinRandomRoom (Hashtable expectedCustomRoomProperties, byte expectedMaxPlayers) [static]`

Attempts to join an open room with fitting, custom properties but fails if none is currently available.

Rooms can be created in arbitrary lobbies which get created on demand. You can join rooms from any lobby without actually joining the lobby. Use the `JoinRandomRoom` overload with [TypedLobby](#) parameter.

This method will only match rooms attached to one lobby! If you use many lobbies, you might have to repeat `JoinRandomRoom`, to find some fitting room. This method looks up a room in the currently active lobby or (if no lobby is joined) in the default lobby.

If this fails, you can still create a room (and make this available for the next who uses `JoinRandomRoom`). Alternatively, try again in a moment.

Parameters

<i>expectedCustomRoomProperties</i>	Filters for rooms that match these custom properties (string keys and values). To ignore, pass null.
<i>expectedMaxPlayers</i>	Filters for a particular maxplayer setting. Use 0 to accept any maxPlayer value.

8.40.2.32 `static bool PhotonNetwork.JoinRandomRoom (Hashtable expectedCustomRoomProperties, byte expectedMaxPlayers, MatchmakingMode matchingType, TypedLobby typedLobby, string sqlLobbyFilter) [static]`

Attempts to join an open room with fitting, custom properties but fails if none is currently available.

Rooms can be created in arbitrary lobbies which get created on demand. You can join rooms from any lobby without actually joining the lobby with this overload.

This method will only match rooms attached to one lobby! If you use many lobbies, you might have to repeat Join↔RandomRoom, to find some fitting room. This method looks up a room in the specified lobby or the currently active lobby (if none specified) or in the default lobby (if none active).

If this fails, you can still create a room (and make this available for the next who uses JoinRandomRoom). Alternatively, try again in a moment.

In offlineMode, a room will be created but no properties will be set and all parameters of this JoinRandomRoom call are ignored. The event/callback OnJoinedRoom gets called (see enum PhotonNetworkingMessage).

Parameters

<i>expectedCustomRoomProperties</i>	Filters for rooms that match these custom properties (string keys and values). To ignore, pass null.
<i>expectedMaxPlayers</i>	Filters for a particular maxplayer setting. Use 0 to accept any maxPlayer value.
<i>matchingType</i>	Selects one of the available matchmaking algorithms. See MatchmakingMode enum for options.
<i>typedLobby</i>	The lobby in which you want to lookup a room. Pass null, to use the default lobby. This does not join that lobby and neither sets the lobby property.
<i>sqlLobbyFilter</i>	A filter-string for SQL-typed lobbies.

8.40.2.33 `static bool PhotonNetwork.JoinRoom (string roomName) [static]`

Join room by roomname and on success calls [OnJoinedRoom\(\)](#).

This is not affected by lobbies.

On success, the method [OnJoinedRoom\(\)](#) is called on any script. You can implement it to react to joining a room.

JoinRoom fails if the room is either full or no longer available (it might become empty while you attempt to join). Implement [OnPhotonJoinRoomFailed\(\)](#) to get a callback in error case.

To join a room from the lobby's listing, use [RoomInfo.name](#) as roomName here. Despite using multiple lobbies, a roomName is always "global" for your application and so you don't have to specify which lobby it's in. The Master Server will find the room. In the [Photon](#) Cloud, an application is defined by AppId, Game- and PUN-version.

[PhotonNetworkingMessage.OnPhotonJoinRoomFailed](#) [PhotonNetworkingMessage.OnJoinedRoom](#)

Parameters

<i>roomName</i>	Unique name of the room to join.
-----------------	----------------------------------

8.40.2.34 static bool PhotonNetwork.LeaveLobby () [static]

Leave a lobby to stop getting updates about available rooms.

This does not reset [PhotonNetwork.lobby](#)! This allows you to join this particular lobby later easily.

The values `countOfPlayers`, `countOfPlayersOnMaster`, `countOfPlayersInRooms` and `countOfRooms` are received even without being in a lobby.

You can use `JoinRandomRoom` without being in a lobby. Use `autoJoinLobby` to not join a lobby when you connect.

8.40.2.35 static bool PhotonNetwork.LeaveRoom () [static]

Leave the current room and return to the Master Server where you can join or create rooms (see remarks).

This will clean up all (network) GameObjects with a [PhotonView](#), unless you changed `autoCleanUp` to false. Returns to the Master Server.

In `OfflineMode`, the local "fake" room gets cleaned up and `OnLeftRoom` gets called immediately.

8.40.2.36 static void PhotonNetwork.LoadLevel (int levelNumber) [static]

Wraps loading a level to pause the network message-queue.

Optionally syncs the loaded level in a room.

To sync the loaded level in a room, set [PhotonNetwork.automaticallySyncScene](#) to true. The Master Client of a room will then sync the loaded level with every other player in the room.

While loading levels, it makes sense to not dispatch messages received by other players. This method takes care of that by setting [PhotonNetwork.isMessageQueueRunning](#) = false and enabling the queue when the level was loaded.

You should make sure you don't fire RPCs before you load another scene (which doesn't contain the same Game↔Objects and PhotonViews). You can call this in `OnJoinedRoom`.

This uses `Application.LoadLevel`.

Parameters

<i>levelNumber</i>	Number of the level to load. When using level numbers, make sure they are identical on all clients.
--------------------	---

8.40.2.37 static void PhotonNetwork.LoadLevel (string levelName) [static]

Wraps loading a level to pause the network message-queue.

Optionally syncs the loaded level in a room.

While loading levels, it makes sense to not dispatch messages received by other players. This method takes care of that by setting `PhotonNetwork.isMessageQueueRunning` = false and enabling the queue when the level was loaded.

To sync the loaded level in a room, set `PhotonNetwork.automaticallySyncScene` to true. The Master Client of a room will then sync the loaded level with every other player in the room.

You should make sure you don't fire RPCs before you load another scene (which doesn't contain the same GameObjects and PhotonViews). You can call this in `OnJoinedRoom`.

This uses `Application.LoadLevel`.

Parameters

<i>levelName</i>	Name of the level to load. Make sure it's available to all clients in the same room.
------------------	--

8.40.2.38 `static void PhotonNetwork.NetworkStatisticsReset () [static]`

Resets the traffic stats and re-enables them.

8.40.2.39 `static string PhotonNetwork.NetworkStatisticsToString () [static]`

Only available when `NetworkStatisticsEnabled` was used to gather some stats.

Returns

A string with vital networking statistics.

8.40.2.40 `static void PhotonNetwork.OverrideBestCloudServer (CloudRegionCode region) [static]`

Overwrites the region that is used for `ConnectToBestCloudServer(string gameVersion)`.

This will overwrite the result of pinging all cloud servers.

Use this to allow your users to save a manually selected region in the player preferences.

Note: You can also use `PhotonNetwork.ConnectToRegion` to (temporarily) connect to a specific region.

8.40.2.41 `static bool PhotonNetwork.RaiseEvent (byte eventCode, object eventContent, bool sendReliable, RaiseEventOptions options) [static]`

Sends fully customizable events in a room.

Events consist of at least an `EventCode` (0..199) and can have content.

To receive the events someone sends, register your handling method in `PhotonNetwork.OnEventCall`.

Example: `private void OnEventHandler(byte eventCode, object content, int senderId) { Debug.Log("OnEventHandler"); }`

[PhotonNetwork.OnEventCall](#) += this.OnEventHandler;

With the senderId, you can look up the [PhotonPlayer](#) who sent the event. It is best practice to assign a eventCode for each different type of content and action. You have to cast the content.

The eventContent is optional. To be able to send something, it must be a "serializable type", something that the client can turn into a byte[] basically. Most basic types and arrays of them are supported, including Unity's Vector2, Vector3, Quaternion. Transforms or classes some project defines are NOT supported! You can make your own class a "serializable type" by following the example in [CustomTypes.cs](#).

The [RaiseEventOptions](#) have some (less intuitive) combination rules: If you set targetActors (an array of [PhotonPlayer.ID](#) values), the receivers parameter gets ignored. When using event caching, the targetActors, receivers and interestGroup can't be used. Buffered events go to all. When using cachingOption removeFromRoomCache, the eventCode and content are actually not sent but used as filter.

Parameters

<i>eventCode</i>	A byte identifying the type of event. You might want to use a code per action or to signal which content can be expected. Allowed: 0..199.
<i>eventContent</i>	Some serializable object like string, byte, integer, float (etc) and arrays of those. Hashtables with byte keys are good to send variable content.
<i>sendReliable</i>	Makes sure this event reaches all players. It gets acknowledged, which requires bandwidth and it can't be skipped (might add lag in case of loss).
<i>options</i>	Allows more complex usage of events. If null, RaiseEventOptions.Default will be used (which is fine).

Returns

False if event could not be sent

8.40.2.42 `static void PhotonNetwork.RefreshCloudServerRating () [static]`

Pings all cloud servers again to find the one with best ping (currently).

8.40.2.43 `static void PhotonNetwork.RemovePlayerCustomProperties (string[] customPropertiesToDelete) [static]`

Locally removes Custom Properties of "this" player.

Important: This does not synchronize the change! Useful when you switch rooms.

Use this method with care. It can create inconsistencies of state between players! This only changes the player's customProperties locally. This can be useful to clear your Custom Properties between games (let's say they store which turn you made, kills, etc).

[SetPlayerCustomProperties\(\)](#) syncs and can be used to set values to null while in a room. That can be considered "removed" while in a room.

If customPropertiesToDelete is null or has 0 entries, all Custom Properties are deleted (replaced with a new Hashtable). If you specify keys to remove, those will be removed from the Hashtable but other keys are unaffected.

Parameters

<i>customPropertiesToDelete</i>	List of Custom Property keys to remove. See remarks.
---------------------------------	--

8.40.2.44 static void PhotonNetwork.RemoveRPCs (PhotonPlayer *targetPlayer*) [static]

Remove all buffered RPCs from server that were sent by targetPlayer.

Can only be called on local player (for "self") or Master Client (for anyone).

This method requires either:

- This is the targetPlayer's client.
- This client is the Master Client (can remove any PhotonPlayer's RPCs).

If the targetPlayer calls RPCs at the same time that this is called, network lag will determine if those get buffered or cleared like the rest.

Parameters

<i>targetPlayer</i>	This player's buffered RPCs get removed from server buffer.
---------------------	---

8.40.2.45 static void PhotonNetwork.RemoveRPCs (PhotonView *targetPhotonView*) [static]

Remove all buffered RPCs from server that were sent via targetPhotonView.

The Master Client and the owner of the targetPhotonView may call this.

This method requires either:

- The targetPhotonView is owned by this client (Instantiated by it).
- This client is the Master Client (can remove any PhotonView's RPCs).

Parameters

<i>targetPhotonView</i>	RPCs buffered for this PhotonView get removed from server buffer.
-------------------------	---

8.40.2.46 static void PhotonNetwork.RemoveRPCsInGroup (int *targetGroup*) [static]

Remove all buffered RPCs from server that were sent in the targetGroup, if this is the Master Client or if this controls the individual PhotonView.

This method requires either:

- This client is the Master Client (can remove any RPCs per group).
- Any other client: each [PhotonView](#) is checked if it is under this client's control. Only those RPCs are removed.

Parameters

<i>targetGroup</i>	Interest group that gets all RPCs removed.
--------------------	--

8.40.2.47 static void PhotonNetwork.SendOutgoingCommands () [static]

Can be used to immediately send the RPCs and Instantiates just called, so they are on their way to the other players.

This could be useful if you do a RPC to load a level and then load it yourself. While loading, no RPCs are sent to others, so this would delay the "load" RPC. You can send the RPC to "others", use this method, disable the message queue (by `isMessageQueueRunning`) and then load.

8.40.2.48 static void PhotonNetwork.SetLevelPrefix (short prefix) [static]

Sets level prefix for PhotonViews instantiated later on.

Don't set it if you need only one!

Important: If you don't use multiple level prefixes, simply don't set this value. The default value is optimized out of the traffic.

This won't affect existing PhotonViews (they can't be changed yet for existing PhotonViews).

Messages sent with a different level prefix will be received but not executed. This affects RPCs, Instantiates and synchronization.

Be aware that PUN never resets this value, you'll have to do so yourself.

Parameters

<i>prefix</i>	Max value is <code>short.MaxValue = 32767</code>
---------------	--

8.40.2.49 static bool PhotonNetwork.SetMasterClient (PhotonPlayer masterClientPlayer) [static]

Asks the server to assign another player as Master Client of your current room.

RPCs and `RaiseEvent` have the option to send messages only to the Master Client of a room. `SetMasterClient` affects which client gets those messages.

This method calls an operation on the server to set a new Master Client, which takes a roundtrip. In case of success, this client and the others get the new Master Client from the server.

`SetMasterClient` tells the server which current Master Client should be replaced with the new one. It will fail, if anything switches the Master Client moments earlier. There is no callback for this error. All clients should get the new Master Client assigned by the server anyways.

See also: [PhotonNetwork.masterClient](#)

On v3 servers: The ReceiverGroup.MasterClient (usable in RPCs) is not affected by this (still points to lowest player.ID in room). Avoid using this enum value (and send to a specific player instead).

If the current Master Client leaves, PUN will detect a new one by "lowest player ID". Implement OnMasterClientSwitched to get a callback in this case. The PUN-selected Master Client might assign a new one.

Make sure you don't create an endless loop of Master-assigning! When selecting a custom Master Client, all clients should point to the same player, no matter who actually assigns this player.

Locally the Master Client is immediately switched, while remote clients get an event. This means the game is temporarily without Master Client like when a current Master Client leaves.

When switching the Master Client manually, keep in mind that this user might leave and not do it's work, just like any Master Client.

Parameters

<i>masterClientPlayer</i>	The player to become the next Master Client.
---------------------------	--

Returns

False when this operation couldn't be done. Must be in a room (not in offlineMode).

8.40.2.50 static void PhotonNetwork.SetPlayerCustomProperties (Hashtable *customProperties*) [static]

Sets this (local) player's properties and synchronizes them to the other players (don't modify them directly).

While in a room, your properties are synced with the other players. CreateRoom, JoinRoom and JoinRandomRoom will all apply your player's custom properties when you enter the room. The whole Hashtable will get sent. Minimize the traffic by setting only updated key/values.

If the Hashtable is null, the custom properties will be cleared. Custom properties are never cleared automatically, so they carry over to the next room, if you don't change them.

Don't set properties by modifying PhotonNetwork.player.customProperties!

Parameters

<i>customProperties</i>	Only string-typed keys will be used from this hashtable. If null, custom properties are all deleted.
-------------------------	--

8.40.2.51 static void PhotonNetwork.SetReceivingEnabled (int *group*, bool *enabled*) [static]

Enable/disable receiving on given group (applied to PhotonViews)

Parameters

<i>group</i>	The interest group to affect.
<i>enabled</i>	Sets if receiving from group to enabled (or not).

8.40.2.52 `static void PhotonNetwork.SetReceivingEnabled (int[] enableGroups, int[] disableGroups) [static]`

Enable/disable receiving on given groups (applied to PhotonViews)

Parameters

<i>enableGroups</i>	The interest groups to enable (or null).
<i>disableGroups</i>	The interest groups to disable (or null).

8.40.2.53 `static void PhotonNetwork.SetSendingEnabled (int group, bool enabled) [static]`

Enable/disable sending on given group (applied to PhotonViews)

Parameters

<i>group</i>	The interest group to affect.
<i>enabled</i>	Sets if sending to group is enabled (or not).

8.40.2.54 `static void PhotonNetwork.SetSendingEnabled (int[] enableGroups, int[] disableGroups) [static]`

Enable/disable sending on given groups (applied to PhotonViews)

Parameters

<i>enableGroups</i>	The interest groups to enable sending on (or null).
<i>disableGroups</i>	The interest groups to disable sending on (or null).

8.40.2.55 `static void PhotonNetwork.SwitchToProtocol (ConnectionProtocol cp) [static]`

While offline, the network protocol can be switched (which affects the ports you can use to connect).

When you switch the protocol, make sure to also switch the port for the master server. Default ports are: TCP: 4530
UDP: 5055

This could look like this:

```
Connect(serverAddress, <udpport|tcpport>, applID, gameVersion)
```

Or when you use [ConnectUsingSettings\(\)](#), the PORT in the settings can be switched like so:

```
PhotonNetwork.PhotonServerSettings.ServerPort = 4530;
```

The current protocol can be read this way:

```
PhotonNetwork.networkingPeer.UsedProtocol
```

This does not work with the native socket plugin of PUN+ on mobile!

Parameters

<i>cp</i>	Network protocol to use as low level connection. UDP is default. TCP is not available on all platforms (see remarks).
-----------	---

8.40.2.56 `static void PhotonNetwork.UnAllocateViewID (int viewID) [static]`

Unregister a viewID (of manually instantiated and destroyed networked objects).

Parameters

<i>viewID</i>	A viewID manually allocated by this player.
---------------	---

8.40.2.57 `static bool PhotonNetwork.WebRpc (string name, object parameters) [static]`

This operation makes [Photon](#) call your custom web-service by name (path) with the given parameters.

This is a server-side feature which must be setup in the [Photon](#) Cloud Dashboard prior to use.

See the Turnbased Feature Overview for a short intro.

<http://doc.photonengine.com/en/turnbased/current/getting-started/feature-overview>
br/> The Parameters will be converted into JSON format, so make sure your parameters are compatible.

See [PhotonNetworkingMessage.OnWebRpcResponse](#) on how to get a response.

It's important to understand that the OperationResponse only tells if the WebRPC could be called. The content of the response contains any values your web-service sent and the error/success code. In case the web-service failed, an error code and a debug message are usually inside the OperationResponse.

The class [WebRpcResponse](#) is a helper-class that extracts the most valuable content from the WebRPC response.

Example callback implementation:

```
public void OnWebRpcResponse(OperationResponse response)
{
    WebRpcResponse webResponse = new WebRpcResponse(operationResponse);
    if (webResponse.ReturnCode != 0) { //...
    }

    switch (webResponse.Name) { //...
    }
    // and so on
}
```


8.40.3 Member Data Documentation

8.40.3.1 float PhotonNetwork.BackgroundTimeout = 0.0f [static]

Defines after how many seconds PUN will close a connection, after Unity's OnApplicationPause(true) call.

The value is set in seconds. Set a value greater than 0.001f, if you want to disconnect in background. Default: 0.0f.

Note: Some platforms (e.g. iOS) don't allow to keep a connection while the app is in background. In those cases, this value does not change anything.

Unity's OnApplicationPause() callback is broken in some exports (Android) of some Unity versions. Make sure OnApplicationPause() gets the callbacks you'd expect on the platform you target! Check PhotonHandler.OnApplicationPause(bool pause), to see the implementation.

8.40.3.2 bool PhotonNetwork.InstantiateInRoomOnly = true [static]

If true, Instantiate methods will check if you are in a room and fail if you are not.

Instantiating anything outside of a specific room is very likely to break things. Turn this off only if you know what you do.

8.40.3.3 PhotonLogLevel PhotonNetwork.logLevel = PhotonLogLevel.ErrorsOnly [static]

Network log level.

Controls how verbose PUN is.

8.40.3.4 readonly int PhotonNetwork.MAX_VIEW_IDS = 1000 [static]

The maximum number of assigned PhotonViews *per player* (or scene).

See the General Documentation topic "Limitations" on how to raise this limitation.

8.40.3.5 int PhotonNetwork.maxConnections [static]

Only used in Unity Networking. In PUN, set the number of players in [PhotonNetwork.CreateRoom](#).

8.40.3.6 EventCallback PhotonNetwork.OnEventCall [static]

Register your RaiseEvent handling methods here by using "+=".

Any eventCode < 200 will be forwarded to your delegate(s).

[RaiseEvent](#)

8.40.3.7 `ServerSettings PhotonNetwork.PhotonServerSettings = (ServerSettings)Resources.Load(PhotonNetwork.serverSettingsAssetFile, typeof(ServerSettings))` [static]

Serialized server settings, written by the Setup Wizard for use in ConnectUsingSettings.

8.40.3.8 `float PhotonNetwork.precisionForFloatSynchronization = 0.01f` [static]

The minimum difference between floats before we send it via a [PhotonView](#)'s OnSerialize/ObservingComponent.

8.40.3.9 `float PhotonNetwork.precisionForQuaternionSynchronization = 1.0f` [static]

The minimum angle that a rotation needs to change before we send it via a [PhotonView](#)'s OnSerialize/ObservingComponent.

8.40.3.10 `float PhotonNetwork.precisionForVectorSynchronization = 0.000099f` [static]

The minimum difference that a Vector2 or Vector3(e.g.

a transforms rotation) needs to change before we send it via a [PhotonView](#)'s OnSerialize/ObservingComponent.

Note that this is the sqrMagnitude. E.g. to send only after a 0.01 change on the Y-axis, we use $0.01f * 0.01f = 0.0001f$. As a remedy against float inaccuracy we use 0.000099f instead of 0.0001f.

8.40.3.11 `Dictionary<string, GameObject> PhotonNetwork.PrefabCache = new Dictionary<string, GameObject>()` [static]

Keeps references to GameObjects for frequent instantiation (out of memory instead of loading the Resources).

You should be able to modify the cache anytime you like, except while Instantiate is used. Best do it only in the main-Thread.

8.40.3.12 `HashSet<GameObject> PhotonNetwork.SendMonoMessageTargets` [static]

If not null, this is the (exclusive) list of GameObjects that get called by PUN SendMonoMessage().

For all callbacks defined in PhotonNetworkingMessage, PUN will use SendMonoMessage and call FindObjectsOfType() to find all scripts and GameObjects that might want a callback by PUN.

PUN callbacks are not very frequent (in-game, property updates are most frequent) but FindObjectsOfType is time consuming and with a large number of GameObjects, performance might suffer.

Optionally, SendMonoMessageTargets can be used to supply a list of target GameObjects. This skips the FindObjectsOfType() but any GameObject that needs callbacks will have to Add itself to this list.

If null, the default behaviour is to do a SendMessage on each GameObject with a MonoBehaviour.

8.40.3.13 Type PhotonNetwork.SendMonoMessageTargetType = typeof(MonoBehaviour) [static]

Defines which classes can contain PUN Callback implementations.

This provides the option to optimize your runtime for speed.

The more specific this Type is, the fewer classes will be checked with reflection for callback methods.

8.40.3.14 bool PhotonNetwork.UsePrefabCache = true [static]

While enabled (true), Instantiate uses [PhotonNetwork.PrefabCache](#) to keep game objects in memory (improving instantiation of the same prefab).

Setting UsePrefabCache to false during runtime will not clear PrefabCache but will ignore it right away. You could clean and modify the cache yourself. Read its comments.

8.40.3.15 bool PhotonNetwork.UseRpcMonoBehaviourCache [static]

While enabled, the MonoBehaviours on which we call RPCs are cached, avoiding costly `GetComponent<MonoBehaviour>()` calls.

RPCs are called on the MonoBehaviours of a target [PhotonView](#). Those have to be found via `GetComponent`.

When set this to true, the list of MonoBehaviours gets cached in each [PhotonView](#). You can use `PhotonView.RefreshRpcMonoBehaviourCache()` to manually refresh a [PhotonView](#)'s list of MonoBehaviours on demand (when a new MonoBehaviour gets added to a networked GameObject, e.g.).

8.40.3.16 const string PhotonNetwork.versionPUN = "1.65"

Version number of PUN. Also used in `GameVersion` to separate client version from each other.

8.40.4 Property Documentation

8.40.4.1 AuthenticationValues PhotonNetwork.AuthValues [static], [get], [set]

A user's authentication values used during connect for Custom Authentication with [Photon](#) (and a custom service/community).

Set these before calling `Connect` if you want custom authentication.

If authentication fails for any values, PUN will call your implementation of [OnCustomAuthenticationFailed\(string debugMsg\)](#). See: [PhotonNetworkingMessage.OnCustomAuthenticationFailed](#)

8.40.4.2 bool PhotonNetwork.autoCleanUpPlayerObjects [static], [get], [set]

This setting defines per room, if network-instantiated GameObjects (with [PhotonView](#)) get cleaned up when the creator of it leaves.

This setting is done per room. It can't be changed in the room and it will override the settings of individual clients.

If `room.AutoCleanUp` is enabled in a room, the PUN clients will destroy a player's GameObjects on leave. This includes GameObjects manually instantiated (via RPCs, e.g.). When enabled, the server will clean RPCs, instantiated GameObjects and PhotonViews of the leaving player, too. and Players who join after someone left, won't get the events of that player anymore.

Under the hood, this setting is stored as a Custom [Room](#) Property. Enabled by default.

8.40.4.3 `bool PhotonNetwork.autoJoinLobby` `[static], [get], [set]`

Set in PhotonServerSettings asset.

Defines if the [PhotonNetwork](#) should join the "lobby" when connected to the Master server.

If this is false, [OnConnectedToMaster\(\)](#) will be called when connection to the Master is available. [OnJoinedLobby\(\)](#) will NOT be called if this is false.

Enabled by default.

The room listing will not become available. Rooms can be created and joined (randomly) without joining the lobby (and getting sent the room list).

8.40.4.4 `bool PhotonNetwork.automaticallySyncScene` `[static], [get], [set]`

Defines if all clients in a room should load the same level as the Master Client (if that used [PhotonNetwork.LoadScene\(\)](#)).

To synchronize the loaded level, the Master Client should use [PhotonNetwork.LoadLevel](#). All clients will load the new scene when they get the update or when they join.

Internally, a Custom [Room](#) Property is set for the loaded scene. When a client reads that and is not in the same scene yet, it will immediately pause the Message Queue ([PhotonNetwork.isMessageQueueRunning](#) = false) and load. When the scene finished loading, PUN will automatically re-enable the Message Queue.

8.40.4.5 `bool PhotonNetwork.connected` `[static], [get]`

False until you connected to [Photon](#) initially.

True in offline mode, while connected to any server and even while switching servers.

8.40.4.6 `bool PhotonNetwork.connectedAndReady` `[static], [get]`

A refined version of connected which is true only if your connection to the server is ready to accept operations like join, leave, etc.

8.40.4.7 `bool PhotonNetwork.connecting` `[static], [get]`

True when you called [ConnectUsingSettings](#) (or similar) until the low level connection to [Photon](#) gets established.

8.40.4.8 `ConnectionState PhotonNetwork.connectionState` `[static], [get]`

Simplified connection state

8.40.4.9 PeerState PhotonNetwork.connectionStateDetailed [static], [get]

Detailed connection state (ignorant of PUN, so it can be "disconnected" while switching servers).

In OfflineMode, this is [PeerState.Joined](#) (after create/join) or it is ConnectedToMaster in all other cases.

8.40.4.10 int PhotonNetwork.countOfPlayers [static], [get]

The count of players currently using this application (available on MasterServer in 5sec intervals).

8.40.4.11 int PhotonNetwork.countOfPlayersInRooms [static], [get]

Count of users currently playing your app in some room (sent every 5sec by Master Server).

Use playerList.Count to get the count of players in the room you're in!

8.40.4.12 int PhotonNetwork.countOfPlayersOnMaster [static], [get]

The count of players currently looking for a room (available on MasterServer in 5sec intervals).

8.40.4.13 int PhotonNetwork.countOfRooms [static], [get]

The count of rooms currently in use (available on MasterServer in 5sec intervals).

While inside the lobby you can also check the count of listed rooms as: [PhotonNetwork.GetRoomList\(\).Length](#). Since PUN v1.25 this is only based on the statistic event [Photon](#) sends (counting all rooms).

8.40.4.14 bool PhotonNetwork.CrcCheckEnabled [static], [get], [set]

Crc checks can be useful to detect and avoid issues with broken datagrams. Can be enabled while not connected.

8.40.4.15 bool PhotonNetwork.EnableLobbyStatistics [static], [get], [set]

Set in PhotonServerSettings asset.

Enable to get a list of active lobbies from the Master Server.

Lobby Statistics can be useful if a game uses multiple lobbies and you want to show activity of each to players.

This value is stored in PhotonServerSettings.

[PhotonNetwork.LobbyStatistics](#) is updated when you connect to the Master Server. There is also a callback [PunBehaviour](#).

8.40.4.16 List<FriendInfo> PhotonNetwork.Friends [static], [get], [set]

Read-only list of friends, their online status and the room they are in.

Null until initialized by a FindFriends call.

Do not modify this list! It is internally handled by FindFriends and only available to read the values. The value of FriendsListAge tells you how old the data is in milliseconds.

Don't get this list more often than useful (> 10 seconds). In best case, keep the list you fetch really short. You could (e.g.) get the full list only once, then request a few updates only for friends who are online. After a while (e.g. 1 minute), you can get the full list again (to update online states).

8.40.4.17 int PhotonNetwork.FriendsListAge [static], [get]

Age of friend list info (in milliseconds).

It's 0 until a friend list is fetched.

8.40.4.18 string PhotonNetwork.gameVersion [static], [get], [set]

Version string for your this build.

Can be used to separate incompatible clients. Sent during connect.

This is only sent when you connect so that is also the place you set it usually (e.g. in ConnectUsingSettings).

8.40.4.19 bool PhotonNetwork.inRoom [static], [get]

Is true while being in a room (connectionStateDetailed == [PeerState.Joined](#)).

Many actions can only be executed in a room, like Instantiate or Leave, etc. You can join a room in offline mode, too.

8.40.4.20 bool PhotonNetwork.insideLobby [static], [get]

True while this client is in a lobby.

Implement [IPunCallbacks.OnReceivedRoomListUpdate\(\)](#) for a notification when the list of rooms becomes available or updated.

You are automatically leaving any lobby when you join a room! Lobbies only exist on the Master Server (whereas rooms are handled by Game Servers).

8.40.4.21 bool PhotonNetwork.isMasterClient [static], [get]

Are we the master client?

8.40.4.22 bool PhotonNetwork.isMessageQueueRunning [static], [get], [set]

Can be used to pause dispatching of incoming events (RPCs, Instantiates and anything else incoming).

While `IsMessageQueueRunning == false`, the `OnPhotonSerializeView` calls are not done and nothing is sent by a client. Also, incoming messages will be queued until you re-activate the message queue.

This can be useful if you first want to load a level, then go on receiving data of PhotonViews and RPCs. The client will go on receiving and sending acknowledgements for incoming packages and your RPCs/Events. This adds "lag" and can cause issues when the pause is longer, as all incoming messages are just queued.

8.40.4.23 bool PhotonNetwork.isNonMasterClientInRoom [static], [get]

True if we are in a room (client) and NOT the room's masterclient

8.40.4.24 TypedLobby PhotonNetwork.lobby [static], [get], [set]

The lobby that will be used when PUN joins a lobby or creates a game.

The default lobby uses an empty string as name. PUN will enter a lobby on the Master Server if `autoJoinLobby` is set to true. So when you connect or leave a room, PUN automatically gets you into a lobby again.

Check [PhotonNetwork.insideLobby](#) if the client is in a lobby. (`masterServerAndLobby`)

8.40.4.25 List<TypedLobbyInfo> PhotonNetwork.LobbyStatistics [static], [get]

If turned on, the Master Server will provide information about active lobbies for this application.

Lobby Statistics can be useful if a game uses multiple lobbies and you want to show activity of each to players. Per lobby, you get: name, type, room- and player-count.

[PhotonNetwork.LobbyStatistics](#) is updated when you connect to the Master Server. There is also a callback `PunBehaviour.OnLobbyStatisticsUpdate`, which you should implement to update your UI (e.g.).

Lobby Statistics are not turned on by default. Enable them in the `PhotonServerSettings` file of the project.

8.40.4.26 PhotonPlayer PhotonNetwork.masterClient [static], [get]

The Master Client of the current room or null (outside of rooms).

Can be used as "authoritative" client/player to make decisions, run AI or other.

If the current Master Client leaves the room (leave/disconnect), the server will quickly assign someone else. If the current Master Client times out (closed app, lost connection, etc), messages sent to this client are effectively lost for the others! A timeout can take 10 seconds in which no Master Client is active.

Implement the method [IPunCallbacks.OnMasterClientSwitched](#) to be called when the Master Client switched.

Use [PhotonNetwork.SetMasterClient](#), to switch manually to some other player / client.

With `offlineMode == true`, this always returns the [PhotonNetwork.player](#).

8.40.4.27 `int PhotonNetwork.MaxResendsBeforeDisconnect` `[static], [get], [set]`

Defines the number of times a reliable message can be resent before not getting an ACK for it will trigger a disconnect.

Default: 5.

Less resends mean quicker disconnects, while more can lead to much more lag without helping. Min: 3. Max: 10.

8.40.4.28 `bool PhotonNetwork.NetworkStatisticsEnabled` `[static], [get], [set]`

Enables or disables the collection of statistics about this client's traffic.

If you encounter issues with clients, the traffic stats are a good starting point to find solutions. Only with enabled stats, you can use `GetVitalStats`

8.40.4.29 `bool PhotonNetwork.offlineMode` `[static], [get], [set]`

Offline mode can be set to re-use your multiplayer code in singleplayer game modes.

When this is on [PhotonNetwork](#) will not create any connections and there is near to no overhead. Mostly usefull for reusing RPC's and [PhotonNetwork.Instantiate](#)

8.40.4.30 `PhotonPlayer [] PhotonNetwork.otherPlayers` `[static], [get]`

The list of players in the current room, excluding the local player.

This list is only valid, while the client is in a room. It automatically gets updated when someone joins or leaves.

This can be used to list all other players in a room. Each player's [PhotonPlayer.customProperties](#) are accessible (set and synchronized via [PhotonPlayer.SetCustomProperties](#)).

You can use a [PhotonPlayer.TagObject](#) to store an arbitrary object for reference. That is not synchronized via the network.

8.40.4.31 `int PhotonNetwork.PacketLossByCrcCheck` `[static], [get]`

If `CrcCheckEnabled`, this counts the incoming packages that don't have a valid CRC checksum and got rejected.

8.40.4.32 `PhotonPlayer PhotonNetwork.player` `[static], [get]`

The local [PhotonPlayer](#).

Always available and represents this player. `CustomProperties` can be set before entering a room and will be synced as well.

8.40.4.33 PhotonPlayer [] PhotonNetwork.playerList [static],[get]

The list of players in the current room, including the local player.

This list is only valid, while the client is in a room. It automatically gets updated when someone joins or leaves.

This can be used to list all players in a room. Each player's [PhotonPlayer.customProperties](#) are accessible (set and synchronized via [PhotonPlayer.SetCustomProperties](#)).

You can use a [PhotonPlayer.TagObject](#) to store an arbitrary object for reference. That is not synchronized via the network.

8.40.4.34 string PhotonNetwork.playerName [static],[get],[set]

Set to synchronize the player's nickname with everyone in the room(s) you enter.

This sets [PhotonPlayer.name](#).

The playerName is just a nickname and does not have to be unique or backed up with some account.

Set the value any time (e.g. before you connect) and it will be available to everyone you play with.

Access the names of players by: [PhotonPlayer.name](#).

[PhotonNetwork.otherPlayers](#) is a list of other players - each contains the playerName the remote player set.

8.40.4.35 IPunPrefabPool PhotonNetwork.PrefabPool [static],[get],[set]

An Object Pool can be used to keep and reuse instantiated object instances.

It replaced Unity's default Instantiate and Destroy methods.

To use a GameObject pool, implement [IPunPrefabPool](#) and assign it here. Prefabs are identified by name.

8.40.4.36 int PhotonNetwork.QuickResends [static],[get],[set]

In case of network loss, reliable messages can be repeated quickly up to 3 times.

When reliable messages get lost more than once, subsequent repeats are delayed a bit to allow the network to recover.

With this option, the repeats 2 and 3 can be sped up. This can help avoid timeouts but also it increases the speed in which gaps are closed.

When you set this, increase [PhotonNetwork.MaxResendsBeforeDisconnect](#) to 6 or 7.

8.40.4.37 int PhotonNetwork.ResentReliableCommands [static],[get]

Count of commands that got repeated (due to local repeat-timing before an ACK was received).

If this value increases a lot, there is a good chance that a timeout disconnect will happen due to bad conditions.

8.40.4.38 Room PhotonNetwork.room [static], [get]

Get the room we're currently in.

Null if we aren't in any room.

8.40.4.39 int PhotonNetwork.sendRate [static], [get], [set]

Defines how many times per second [PhotonNetwork](#) should send a package.

If you change this, do not forget to also change 'sendRateOnSerialize'.

Less packages are less overhead but more delay. Setting the sendRate to 50 will create up to 50 packages per second (which is a lot!). Keep your target platform in mind: mobile networks are slower and less reliable.

8.40.4.40 int PhotonNetwork.sendRateOnSerialize [static], [get], [set]

Defines how many times per second OnPhotonSerialize should be called on PhotonViews.

Choose this value in relation to [PhotonNetwork.sendRate](#). OnPhotonSerialize will create updates and messages to be sent.

A lower rate takes up less performance but will cause more lag.

8.40.4.41 ServerConnection PhotonNetwork.Server [static], [get]

The server (type) this client is currently connected or connecting to.

[Photon](#) uses 3 different roles of servers: Name Server, Master Server and Game Server.

8.40.4.42 string PhotonNetwork.ServerAddress [static], [get]

Currently used server address (no matter if master or game server).

8.40.4.43 int PhotonNetwork.ServerTimestamp [static], [get]

The current server's millisecond timestamp.

This can be useful to sync actions and events on all clients in one room. The timestamp is based on the server's Environment.TickCount.

It will overflow from a positive to a negative value every so often, so be careful to use only time-differences to check the time delta when things happen.

This is the basis for [PhotonNetwork.time](#).

8.40.4.44 `double PhotonNetwork.time` `[static], [get]`

[Photon](#) network time, synched with the server.

v1.55

This time value depends on the server's `Environment.TickCount`. It is different per server but inside a [Room](#), all clients should have the same value (Rooms are on one server only).

This is not a `DateTime`!

Use this value with care:

It can start with any positive value.

It will "wrap around" from 4294967.295 to 0!

8.40.4.45 `int PhotonNetwork.unreliableCommandsLimit` `[static], [get], [set]`

Used once per dispatch to limit unreliable commands per channel (so after a pause, many channels can still cause a lot of unreliable commands)

The documentation for this class was generated from the following file:

- `C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/Photon↔ Network/PhotonNetwork.cs`

8.41 PhotonPingManager Class Reference

Public Member Functions

- `IEnumerator` [PingSocket](#) ([Region](#) region)

Static Public Member Functions

- static string [ResolveHost](#) (string hostName)
Attempts to resolve a hostname into an IP string or returns empty string if that fails.

Public Attributes

- bool [UseNative](#)

Static Public Attributes

- static int [Attempts](#) = 5
- static bool [IgnoreInitialAttempt](#) = true
- static int [MaxMillisecondsPerPing](#) = 800

Properties

- [Region BestRegion](#) [get]
- bool [Done](#) [get]

8.41.1 Member Function Documentation

8.41.1.1 IEnumerator PhotonPingManager.PingSocket ([Region region](#))

Affected by frame-rate of app, as this Coroutine checks the socket for a result once per frame.

8.41.1.2 static string PhotonPingManager.ResolveHost ([string hostName](#)) [static]

Attempts to resolve a hostname into an IP string or returns empty string if that fails.

Parameters

<i>hostName</i>	Hostname to resolve.
-----------------	----------------------

Returns

IP string or empty string if resolution fails

8.41.2 Member Data Documentation

8.41.2.1 int PhotonPingManager.Attempts = 5 [static]

8.41.2.2 bool PhotonPingManager.IgnoreInitialAttempt = true [static]

8.41.2.3 int PhotonPingManager.MaxMillisecondsPerPing = 800 [static]

8.41.2.4 bool PhotonPingManager.UseNative

8.41.3 Property Documentation

8.41.3.1 [Region](#) PhotonPingManager.BestRegion [get]

8.41.3.2 bool PhotonPingManager.Done [get]

The documentation for this class was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/Photon↔ Network/[PingCloudRegions.cs](#)

8.42 PhotonPlayer Class Reference

Summarizes a "player" within a room, identified (in that room) by actorID.

Public Member Functions

- [PhotonPlayer](#) (bool [isLocal](#), int actorID, string [name](#))
Creates a [PhotonPlayer](#) instance.
- override bool [Equals](#) (object p)
Makes [PhotonPlayer](#) comparable
- override int [GetHashCode](#) ()
- void [SetCustomProperties](#) ([Hashtable](#) propertiesToSet, [Hashtable](#) expectedValues=null, bool web↔
Forward=false)
Updates the this player's Custom Properties with new/updated key-values.
- [PhotonPlayer](#) [Get](#) (int id)
- [PhotonPlayer](#) [GetNext](#) ()
- [PhotonPlayer](#) [GetNextFor](#) ([PhotonPlayer](#) currentPlayer)
- [PhotonPlayer](#) [GetNextFor](#) (int currentPlayerId)
- override string [ToString](#) ()
Brief summary string of the [PhotonPlayer](#).
- string [ToStringFull](#) ()
String summary of the [PhotonPlayer](#): player.ID, name and all custom properties of this user.

Static Public Member Functions

- static [PhotonPlayer](#) [Find](#) (int [ID](#))
Try to get a specific player by id.

Public Attributes

- readonly bool [isLocal](#) = false
Only one player is controlled by each client. Others are not local.
- object [TagObject](#)
Can be used to store a reference that's useful to know "by player".

Protected Member Functions

- [PhotonPlayer](#) (bool [isLocal](#), int actorID, [Hashtable](#) properties)
Internally used to create players from event Join

Properties

- int [ID](#) [get]
This player's actorID
- string [name](#) [get, set]
Nickname of this player.
- bool [isMasterClient](#) [get]
True if this player is the Master Client of the current room.
- [Hashtable](#) [customProperties](#) [get, set]
Read-only cache for custom properties of player.
- [Hashtable](#) [allProperties](#) [get]
Creates a Hashtable with all properties (custom and "well known" ones).

8.42.1 Detailed Description

Summarizes a "player" within a room, identified (in that room) by actorID.

Each player has an actorId (or ID), valid for that room. It's -1 until it's assigned by server. Each client can set it's player's custom properties with SetCustomProperties, even before being in a room. They are synced when joining a room.

8.42.2 Constructor & Destructor Documentation

8.42.2.1 PhotonPlayer.PhotonPlayer (bool *isLocal*, int *actorID*, string *name*)

Creates a [PhotonPlayer](#) instance.

Parameters

<i>isLocal</i>	If this is the local peer's player (or a remote one).
<i>actorID</i>	ID or ActorNumber of this player in the current room (a shortcut to identify each player in room)
<i>name</i>	Name of the player (a "well known property").

8.42.2.2 PhotonPlayer.PhotonPlayer (bool *isLocal*, int *actorID*, Hashtable *properties*) [protected]

Internally used to create players from event Join

8.42.3 Member Function Documentation

8.42.3.1 override bool PhotonPlayer.Equals (object *p*)

Makes [PhotonPlayer](#) comparable

8.42.3.2 static PhotonPlayer PhotonPlayer.Find (int *ID*) [static]

Try to get a specific player by id.

Parameters

<i>ID</i>	ActorID
-----------	---------

Returns

The player with matching actorID or null, if the actorID is not in use.

8.42.3.3 PhotonPlayer PhotonPlayer.Get (int *id*)

8.42.3.4 `override int PhotonPlayer.GetHashCode ()`

8.42.3.5 `PhotonPlayer PhotonPlayer.GetNext ()`

8.42.3.6 `PhotonPlayer PhotonPlayer.GetNextFor (PhotonPlayer currentPlayer)`

8.42.3.7 `PhotonPlayer PhotonPlayer.GetNextFor (int currentPlayerId)`

8.42.3.8 `void PhotonPlayer.SetCustomProperties (Hashtable propertiesToSet, Hashtable expectedValues = null, bool webForward = false)`

Updates the this player's Custom Properties with new/updated key-values.

Custom Properties are a key-value set (Hashtable) which is available to all players in a room. They can relate to the room or individual players and are useful when only the current value of something is of interest. For example: The map of a room. All keys must be strings.

The [Room](#) and the [PhotonPlayer](#) class both have SetCustomProperties methods. Also, both classes offer access to current key-values by: customProperties.

Always use SetCustomProperties to change values. To reduce network traffic, set only values that actually changed. New properties are added, existing values are updated. Other values will not be changed, so only provide values that changed or are new.

To delete a named (custom) property of this room, use null as value.

Locally, SetCustomProperties will update it's cache without delay. Other clients are updated through [Photon](#) (the server) with a fitting operation.

Check and Swap

SetCustomProperties have the option to do a server-side Check-And-Swap (CAS): Values only get updated if the expected values are correct. The expectedValues can be different key/values than the propertiesToSet. So you can check some key and set another key's value (if the check succeeds).

If the client's knowledge of properties is wrong or outdated, it can't set values with CAS. This can be useful to keep players from concurrently setting values. For example: If all players try to pickup some card or item, only one should get it. With CAS, only the first SetProperty gets executed server-side and any other (sent at the same time) fails.

The server will broadcast successfully changed values and the local "cache" of customProperties only gets updated after a roundtrip (if anything changed).

You can do a "webForward": [Photon](#) will send the changed properties to a WebHook defined for your application.

OfflineMode

While [PhotonNetwork.offlineMode](#) is true, the expectedValues and webForward parameters are ignored. In Offline↔ Mode, the local customProperties values are immediately updated (without the roundtrip).

Parameters

<i>propertiesToSet</i>	The new properties to be set.
<i>expectedValues</i>	At least one property key/value set to check server-side. Key and value must be correct. Ignored in OfflineMode.
<i>webForward</i>	Set to true, to forward the set properties to a WebHook, defined for this app (in Dashboard). Ignored in OfflineMode.

8.42.3.9 override string PhotonPlayer.ToString ()

Brief summary string of the [PhotonPlayer](#).

Includes name or player.ID and if it's the Master Client.

8.42.3.10 string PhotonPlayer.ToStringFull ()

String summary of the [PhotonPlayer](#): player.ID, name and all custom properties of this user.

Use with care and not every frame! Converts the customProperties to a String on every single call.

8.42.4 Member Data Documentation

8.42.4.1 readonly bool PhotonPlayer.isLocal = false

Only one player is controlled by each client. Others are not local.

8.42.4.2 object PhotonPlayer.TagObject

Can be used to store a reference that's useful to know "by player".

Example: Set a player's character as Tag by assigning the GameObject on Instantiate.

8.42.5 Property Documentation

8.42.5.1 Hashtable PhotonPlayer.allProperties [get]

Creates a Hashtable with all properties (custom and "well known" ones).

If used more often, this should be cached.

8.42.5.2 Hashtable PhotonPlayer.customProperties [get], [set]

Read-only cache for custom properties of player.

Set via [PhotonPlayer.SetCustomProperties](#).

Don't modify the content of this Hashtable. Use SetCustomProperties and the properties of this class to modify values. When you use those, the client will sync values with the server.

[SetCustomProperties](#)

8.42.5.3 int PhotonPlayer.ID [get]

This player's actorID

8.42.5.4 `bool PhotonPlayer.isMasterClient` `[get]`

True if this player is the Master Client of the current room.

See also: [PhotonNetwork.masterClient](#).

8.42.5.5 `string PhotonPlayer.name` `[get]`, `[set]`

Nickname of this player.

Set the [PhotonNetwork.playerName](#) to make the name synchronized in a room.

The documentation for this class was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/Photon↔ Network/[PhotonPlayer.cs](#)

8.43 PhotonRigidbody2DView Class Reference

This class helps you to synchronize the velocities of a 2d physics Rigidbody.

Inherits MonoBehaviour.

8.43.1 Detailed Description

This class helps you to synchronize the velocities of a 2d physics Rigidbody.

Note that only the velocities are synchronized and because Unitys physics engine is not deterministic (ie. the results aren't always the same on all computers) - the actual positions of the objects may go out of sync. If you want to have the position of this object the same on all clients, you should also add a [PhotonTransformView](#) to synchronize the position. Simply add the component to your GameObject and make sure that the [PhotonRigidbody2DView](#) is added to the list of observed components

The documentation for this class was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/Photon↔ Network/Views/[PhotonRigidbody2DView.cs](#)

8.44 PhotonRigidbody2DViewEditor Class Reference

Inherits Editor.

Public Member Functions

- override void [OnInspectorGUI](#) ()

8.44.1 Member Function Documentation

8.44.1.1 override void PhotonRigidbody2DViewEditor.OnInspectorGUI ()

The documentation for this class was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Editor/Photon↔
Network/Views/[PhotonRigidbody2DViewEditor.cs](#)

8.45 PhotonRigidbodyView Class Reference

This class helps you to synchronize the velocities of a physics Rigidbody.

Inherits MonoBehaviour.

8.45.1 Detailed Description

This class helps you to synchronize the velocities of a physics Rigidbody.

Note that only the velocities are synchronized and because Unitys physics engine is not deterministic (ie. the results aren't always the same on all computers) - the actual positions of the objects may go out of sync. If you want to have the position of this object the same on all clients, you should also add a [PhotonTransformView](#) to synchronize the position. Simply add the component to your GameObject and make sure that the [PhotonRigidbodyView](#) is added to the list of observed components

The documentation for this class was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/Photon↔
Network/Views/[PhotonRigidbodyView.cs](#)

8.46 PhotonRigidbodyViewEditor Class Reference

Inherits Editor.

Public Member Functions

- override void [OnInspectorGUI](#) ()

8.46.1 Member Function Documentation

8.46.1.1 override void PhotonRigidbodyViewEditor.OnInspectorGUI ()

The documentation for this class was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Editor/Photon↔
Network/Views/[PhotonRigidbodyViewEditor.cs](#)

8.47 PhotonStatsGui Class Reference

Basic GUI to show traffic and health statistics of the connection to [Photon](#), toggled by shift+tab.

Inherits MonoBehaviour.

Public Member Functions

- void [Start](#) ()
- void [Update](#) ()
Checks for shift+tab input combination (to toggle statsOn).
- void [OnGUI](#) ()
- void [TrafficStatsWindow](#) (int windowID)

Public Attributes

- bool [statsWindowOn](#) = true
Shows or hides GUI (does not affect if stats are collected).
- bool [statsOn](#) = true
Option to turn collecting stats on or off (used in [Update\(\)](#)).
- bool [healthStatsVisible](#)
Shows additional "health" values of connection.
- bool [trafficStatsOn](#)
Shows additional "lower level" traffic stats.
- bool [buttonsOn](#)
Show buttons to control stats and reset them.
- Rect [statsRect](#) = new Rect(0, 100, 200, 50)
Positioning rect for window.
- int [WindowId](#) = 100
Unity GUI Window ID (must be unique or will cause issues).

8.47.1 Detailed Description

Basic GUI to show traffic and health statistics of the connection to [Photon](#), toggled by shift+tab.

The shown health values can help identify problems with connection losses or performance. Example: If the time delta between two consecutive SendOutgoingCommands calls is a second or more, chances rise for a disconnect being caused by this (because acknowledgements to the server need to be sent in due time).

8.47.2 Member Function Documentation

8.47.2.1 void PhotonStatsGui.OnGUI ()

8.47.2.2 void PhotonStatsGui.Start ()

8.47.2.3 void PhotonStatsGui.TrafficStatsWindow (int windowID)

8.47.2.4 void PhotonStatsGui.Update ()

Checks for shift+tab input combination (to toggle statsOn).

8.47.3 Member Data Documentation

8.47.3.1 `bool PhotonStatsGui.buttonsOn`

Show buttons to control stats and reset them.

8.47.3.2 `bool PhotonStatsGui.healthStatsVisible`

Shows additional "health" values of connection.

8.47.3.3 `bool PhotonStatsGui.statsOn = true`

Option to turn collecting stats on or off (used in [Update\(\)](#)).

8.47.3.4 `Rect PhotonStatsGui.statsRect = new Rect(0, 100, 200, 50)`

Positioning rect for window.

8.47.3.5 `bool PhotonStatsGui.statsWindowOn = true`

Shows or hides GUI (does not affect if stats are collected).

8.47.3.6 `bool PhotonStatsGui.trafficStatsOn`

Shows additional "lower level" traffic stats.

8.47.3.7 `int PhotonStatsGui.WindowId = 100`

Unity GUI Window ID (must be unique or will cause issues).

The documentation for this class was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/Photon↔ Network/[PhotonStatsGui.cs](#)

8.48 PhotonStream Class Reference

This container is used in [OnPhotonSerializeView\(\)](#) to either provide incoming data of a [PhotonView](#) or for you to provide it.

Public Member Functions

- [PhotonStream](#) (bool write, object[] incomingData)
Creates a stream and initializes it.
- object [ReceiveNext](#) ()
Read next piece of data from the stream when isReading is true.
- object [PeekNext](#) ()
Read next piece of data from the stream without advancing the "current" item.
- void [SendNext](#) (object obj)
Add another piece of data to send it when isWriting is true.
- object[] [ToArray](#) ()
Turns the stream into a new object[].
- void [Serialize](#) (ref bool myBool)
Will read or write the value, depending on the stream's isWriting value.
- void [Serialize](#) (ref int myInt)
Will read or write the value, depending on the stream's isWriting value.
- void [Serialize](#) (ref string value)
Will read or write the value, depending on the stream's isWriting value.
- void [Serialize](#) (ref char value)
Will read or write the value, depending on the stream's isWriting value.
- void [Serialize](#) (ref short value)
Will read or write the value, depending on the stream's isWriting value.
- void [Serialize](#) (ref float obj)
Will read or write the value, depending on the stream's isWriting value.
- void [Serialize](#) (ref [PhotonPlayer](#) obj)
Will read or write the value, depending on the stream's isWriting value.
- void [Serialize](#) (ref Vector3 obj)
Will read or write the value, depending on the stream's isWriting value.
- void [Serialize](#) (ref Vector2 obj)
Will read or write the value, depending on the stream's isWriting value.
- void [Serialize](#) (ref Quaternion obj)
Will read or write the value, depending on the stream's isWriting value.

Properties

- bool [isWriting](#) [get]
If true, this client should add data to the stream to send it.
- bool [isReading](#) [get]
If true, this client should read data send by another client.
- int [Count](#) [get]
Count of items in the stream.

8.48.1 Detailed Description

This container is used in [OnPhotonSerializeView\(\)](#) to either provide incoming data of a [PhotonView](#) or for you to provide it.

The `isWriting` property will be true if this client is the "owner" of the [PhotonView](#) (and thus the `GameObject`). Add data to the stream and it's sent via the server to the other players in a room. On the receiving side, `isWriting` is false and the data should be read.

Send as few data as possible to keep connection quality up. An empty [PhotonStream](#) will not be sent.

Use either [Serialize\(\)](#) for reading and writing or [SendNext\(\)](#) and [ReceiveNext\(\)](#). The latter two are just explicit read and write methods but do about the same work as [Serialize\(\)](#). It's a matter of preference which methods you use.

See also

[PhotonNetworkingMessage](#)

8.48.2 Constructor & Destructor Documentation

8.48.2.1 `PhotonStream.PhotonStream (bool write, object[] incomingData)`

Creates a stream and initializes it.

Used by PUN internally.

8.48.3 Member Function Documentation

8.48.3.1 `object PhotonStream.PeekNext ()`

Read next piece of data from the stream without advancing the "current" item.

8.48.3.2 `object PhotonStream.ReceiveNext ()`

Read next piece of data from the stream when `isReading` is true.

8.48.3.3 `void PhotonStream.SendNext (object obj)`

Add another piece of data to send it when `isWriting` is true.

8.48.3.4 `void PhotonStream.Serialize (ref bool myBool)`

Will read or write the value, depending on the stream's `isWriting` value.

8.48.3.5 `void PhotonStream.Serialize (ref int myInt)`

Will read or write the value, depending on the stream's `isWriting` value.

8.48.3.6 void PhotonStream.Serialize (ref string *value*)

Will read or write the value, depending on the stream's isWriting value.

8.48.3.7 void PhotonStream.Serialize (ref char *value*)

Will read or write the value, depending on the stream's isWriting value.

8.48.3.8 void PhotonStream.Serialize (ref short *value*)

Will read or write the value, depending on the stream's isWriting value.

8.48.3.9 void PhotonStream.Serialize (ref float *obj*)

Will read or write the value, depending on the stream's isWriting value.

8.48.3.10 void PhotonStream.Serialize (ref PhotonPlayer *obj*)

Will read or write the value, depending on the stream's isWriting value.

8.48.3.11 void PhotonStream.Serialize (ref Vector3 *obj*)

Will read or write the value, depending on the stream's isWriting value.

8.48.3.12 void PhotonStream.Serialize (ref Vector2 *obj*)

Will read or write the value, depending on the stream's isWriting value.

8.48.3.13 void PhotonStream.Serialize (ref Quaternion *obj*)

Will read or write the value, depending on the stream's isWriting value.

8.48.3.14 object [] PhotonStream.ToArray ()

Turns the stream into a new object[].

8.48.4 Property Documentation**8.48.4.1 int PhotonStream.Count [get]**

Count of items in the stream.

8.48.4.2 bool PhotonStream.isReading [get]

If true, this client should read data send by another client.

8.48.4.3 bool PhotonStream.isWriting [get]

If true, this client should add data to the stream to send it.

The documentation for this class was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/Photon↔ Network/PhotonClasses.cs

8.49 PhotonStreamQueue Class Reference

The [PhotonStreamQueue](#) helps you poll object states at higher frequencies then what [PhotonNetwork.sendRate](#) dictates and then sends all those states at once when [Serialize\(\)](#) is called.

Public Member Functions

- [PhotonStreamQueue](#) (int sampleRate)
Initializes a new instance of the [PhotonStreamQueue](#) class.
- void [Reset](#) ()
Resets the [PhotonStreamQueue](#).
- void [SendNext](#) (object obj)
Adds the next object to the queue.
- bool [HasQueuedObjects](#) ()
Determines whether the queue has stored any objects
- object [ReceiveNext](#) ()
Receives the next object from the queue.
- void [Serialize](#) ([PhotonStream](#) stream)
Serializes the specified stream.
- void [Deserialize](#) ([PhotonStream](#) stream)
Deserializes the specified stream.

8.49.1 Detailed Description

The [PhotonStreamQueue](#) helps you poll object states at higher frequencies then what [PhotonNetwork.sendRate](#) dictates and then sends all those states at once when [Serialize\(\)](#) is called.

On the receiving end you can call [Deserialize\(\)](#) and then the stream will roll out the received object states in the same order and timeStep they were recorded in.

8.49.2 Constructor & Destructor Documentation

8.49.2.1 PhotonStreamQueue.PhotonStreamQueue (int sampleRate)

Initializes a new instance of the [PhotonStreamQueue](#) class.

Parameters

<i>sampleRate</i>	How many times per second should the object states be sampled
-------------------	---

8.49.3 Member Function Documentation

8.49.3.1 void PhotonStreamQueue.Deserialize (PhotonStream stream)

Deserializes the specified stream.

Call this in your OnPhotonSerializeView method to receive the whole recorded stream.

Parameters

<i>stream</i>	The PhotonStream you receive as a parameter in OnPhotonSerializeView
---------------	--

8.49.3.2 bool PhotonStreamQueue.HasQueuedObjects ()

Determines whether the queue has stored any objects

8.49.3.3 object PhotonStreamQueue.ReceiveNext ()

Receives the next object from the queue.

This works just like [PhotonStream.ReceiveNext](#)

Returns

8.49.3.4 void PhotonStreamQueue.Reset ()

Resets the [PhotonStreamQueue](#).

You need to do this whenever the amount of objects you are observing changes

8.49.3.5 void PhotonStreamQueue.SendNext (object obj)

Adds the next object to the queue.

This works just like [PhotonStream.SendNext](#)

Parameters

<i>obj</i>	The object you want to add to the queue
------------	---

8.49.3.6 void PhotonStreamQueue.Serialize (PhotonStream stream)

Serializes the specified stream.

Call this in your OnPhotonSerializeView method to send the whole recorded stream.

Parameters

<i>stream</i>	The PhotonStream you receive as a parameter in OnPhotonSerializeView
---------------	--

The documentation for this class was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/Photon↔ Network/[PhotonStreamQueue.cs](#)

8.50 PhotonTransformView Class Reference

This class helps you to synchronize position, rotation and scale of a GameObject.

Inherits MonoBehaviour, and [IPunObservable](#).

Public Member Functions

- void [SetSynchronizedValues](#) (Vector3 speed, float turnSpeed)
These values are synchronized to the remote objects if the interpolation mode or the extrapolation mode SynchronizeValues is used.
- void [OnPhotonSerializeView](#) ([PhotonStream](#) stream, [PhotonMessageInfo](#) info)
Called by PUN several times per second, so that your script can write and read synchronization data for the [Photon↔ View](#).

8.50.1 Detailed Description

This class helps you to synchronize position, rotation and scale of a GameObject.

It also gives you many different options to make the synchronized values appear smooth, even when the data is only send a couple of times per second. Simply add the component to your GameObject and make sure that the [PhotonTransformView](#) is added to the list of observed components

8.50.2 Member Function Documentation

8.50.2.1 void PhotonTransformView.OnPhotonSerializeView (PhotonStream *stream*, PhotonMessageInfo *info*)

Called by PUN several times per second, so that your script can write and read synchronization data for the [PhotonView](#).

This method will be called in scripts that are assigned as Observed component of a [PhotonView](#). [PhotonNetwork.sendRateOnSerialize](#) affects how often this method is called. [PhotonNetwork.sendRate](#) affects how often packages are sent by this client.

Implementing this method, you can customize which data a [PhotonView](#) regularly synchronizes. Your code defines what is being sent (content) and how your data is used by receiving clients.

Unlike other callbacks, *OnPhotonSerializeView* only gets called when it is assigned to a [PhotonView](#) as [PhotonView.observed](#) script.

To make use of this method, the [PhotonStream](#) is essential. It will be in "writing" mode" on the client that controls a PhotonView (PhotonStream.isWriting == true) and in "reading mode" on the remote clients that just receive that the controlling client sends.

If you skip writing any value into the stream, PUN will skip the update. Used carefully, this can conserve bandwidth and messages (which have a limit per room/second).

Note that OnPhotonSerializeView is not called on remote clients when the sender does not send any update. This can't be used as "x-times per second Update()".

Implements [IPunObservable](#).

8.50.2.2 void PhotonTransformView.SetSynchronizedValues (Vector3 *speed*, float *turnSpeed*)

These values are synchronized to the remote objects if the interpolation mode or the extrapolation mode SynchronizeValues is used.

Your movement script should pass on the current speed (in units/second) and turning speed (in angles/second) so the remote object can use them to predict the objects movement.

Parameters

<i>speed</i>	The current movement vector of the object in units/second.
<i>turnSpeed</i>	The current turn speed of the object in angles/second.

The documentation for this class was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/PhotonNetwork/Views/[PhotonTransformView.cs](#)

8.51 PhotonTransformViewEditor Class Reference

Inherits Editor.

Public Member Functions

- void [OnEnable](#) ()
- override void [OnInspectorGUI](#) ()

8.51.1 Member Function Documentation

8.51.1.1 void [PhotonTransformViewEditor.OnEnable](#) ()

8.51.1.2 override void [PhotonTransformViewEditor.OnInspectorGUI](#) ()

The documentation for this class was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Editor/Photon↔
Network/Views/[PhotonTransformViewEditor.cs](#)

8.52 PhotonTransformViewPositionControl Class Reference

Public Member Functions

- [PhotonTransformViewPositionControl](#) ([PhotonTransformViewPositionModel](#) model)
- void [SetSynchronizedValues](#) (Vector3 speed, float turnSpeed)
These values are synchronized to the remote objects if the interpolation mode or the extrapolation mode SynchronizeValues is used.
- Vector3 [UpdatePosition](#) (Vector3 currentPosition)
Calculates the new position based on the values setup in the inspector
- Vector3 [GetNetworkPosition](#) ()
Gets the last position that was received through the network
- Vector3 [GetExtrapolatedPositionOffset](#) ()
Calculates an estimated position based on the last synchronized position, the time when the last position was received and the movement speed of the object
- void [OnPhotonSerializeView](#) (Vector3 currentPosition, [PhotonStream](#) stream, [PhotonMessageInfo](#) info)

8.52.1 Constructor & Destructor Documentation

8.52.1.1 [PhotonTransformViewPositionControl.PhotonTransformViewPositionControl](#) ([PhotonTransformViewPosition↔
Model](#) model)

8.52.2 Member Function Documentation

8.52.2.1 Vector3 [PhotonTransformViewPositionControl.GetExtrapolatedPositionOffset](#) ()

Calculates an estimated position based on the last synchronized position, the time when the last position was received and the movement speed of the object

Returns

Estimated position of the remote object

8.52.2.2 Vector3 PhotonTransformViewPositionControl.GetNetworkPosition ()

Gets the last position that was received through the network

Returns

8.52.2.3 void PhotonTransformViewPositionControl.OnPhotonSerializeView (Vector3 *currentPosition*, PhotonStream *stream*, PhotonMessageInfo *info*)8.52.2.4 void PhotonTransformViewPositionControl.SetSynchronizedValues (Vector3 *speed*, float *turnSpeed*)

These values are synchronized to the remote objects if the interpolation mode or the extrapolation mode SynchronizeValues is used.

Your movement script should pass on the current speed (in units/second) and turning speed (in angles/second) so the remote object can use them to predict the objects movement.

Parameters

<i>speed</i>	The current movement vector of the object in units/second.
<i>turnSpeed</i>	The current turn speed of the object in angles/second.

8.52.2.5 Vector3 PhotonTransformViewPositionControl.UpdatePosition (Vector3 *currentPosition*)

Calculates the new position based on the values setup in the inspector

Parameters

<i>currentPosition</i>	The current position.
------------------------	-----------------------

Returns

The new position.

The documentation for this class was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/Photon↵
Network/Views/[PhotonTransformViewPositionControl.cs](#)

8.53 PhotonTransformViewPositionModel Class Reference

Public Types

- enum [InterpolateOptions](#) {
 [InterpolateOptions.Disabled](#), [InterpolateOptions.FixedSpeed](#), [InterpolateOptions.EstimatedSpeed](#), [Interpolate↵
Options.SynchronizeValues](#),
 [InterpolateOptions.Lerp](#) }

- enum [ExtrapolateOptions](#) { [ExtrapolateOptions.Disabled](#), [ExtrapolateOptions.SynchronizeValues](#), [ExtrapolateOptions.EstimateSpeedAndTurn](#), [ExtrapolateOptions.FixedSpeed](#) }

Public Attributes

- bool [SynchronizeEnabled](#)
- bool [TeleportEnabled](#) = true
- float [TeleportIfDistanceGreaterThan](#) = 3f
- [InterpolateOptions](#) [InterpolateOption](#) = [InterpolateOptions.EstimatedSpeed](#)
- float [InterpolateMoveTowardsSpeed](#) = 1f
- float [InterpolateLerpSpeed](#) = 1f
- float [InterpolateMoveTowardsAcceleration](#) = 2
- float [InterpolateMoveTowardsDeceleration](#) = 2
- AnimationCurve [InterpolateSpeedCurve](#)
- [ExtrapolateOptions](#) [ExtrapolateOption](#) = [ExtrapolateOptions.Disabled](#)
- float [ExtrapolateSpeed](#) = 1f
- bool [ExtrapolateIncludingRoundTripTime](#) = true
- int [ExtrapolateNumberOfStoredPositions](#) = 1
- bool [DrawErrorGizmo](#) = true

8.53.1 Member Enumeration Documentation

8.53.1.1 enum [PhotonTransformViewPositionModel.ExtrapolateOptions](#) [strong]

Enumerator

Disabled

SynchronizeValues

EstimateSpeedAndTurn

FixedSpeed

8.53.1.2 enum [PhotonTransformViewPositionModel.InterpolateOptions](#) [strong]

Enumerator

Disabled

FixedSpeed

EstimatedSpeed

SynchronizeValues

Lerp

8.53.2 Member Data Documentation

8.53.2.1 `bool PhotonTransformViewPositionModel.DrawErrorGizmo = true`

8.53.2.2 `bool PhotonTransformViewPositionModel.ExtrapolateIncludingRoundTripTime = true`

8.53.2.3 `int PhotonTransformViewPositionModel.ExtrapolateNumberOfStoredPositions = 1`

8.53.2.4 **ExtrapolateOptions** `PhotonTransformViewPositionModel.ExtrapolateOption = ExtrapolateOptions.Disabled`

8.53.2.5 `float PhotonTransformViewPositionModel.ExtrapolateSpeed = 1f`

8.53.2.6 `float PhotonTransformViewPositionModel.InterpolateLerpSpeed = 1f`

8.53.2.7 `float PhotonTransformViewPositionModel.InterpolateMoveTowardsAcceleration = 2`

8.53.2.8 `float PhotonTransformViewPositionModel.InterpolateMoveTowardsDeceleration = 2`

8.53.2.9 `float PhotonTransformViewPositionModel.InterpolateMoveTowardsSpeed = 1f`

8.53.2.10 **InterpolateOptions** `PhotonTransformViewPositionModel.InterpolateOption = InterpolateOptions.↵
EstimatedSpeed`

8.53.2.11 **AnimationCurve** `PhotonTransformViewPositionModel.InterpolateSpeedCurve`

Initial value:

```
= new AnimationCurve( new Keyframe[] {
    new Keyframe( -1, 0, 0, Mathf
    .Infinity ),
    new Keyframe( 0, 1, 0, 0 ),
    new Keyframe( 1, 1, 0, 1 ),
    new Keyframe( 4, 4, 1, 0 ) }
)
```

8.53.2.12 `bool PhotonTransformViewPositionModel.SynchronizeEnabled`

8.53.2.13 `bool PhotonTransformViewPositionModel.TeleportEnabled = true`

8.53.2.14 `float PhotonTransformViewPositionModel.TeleportIfDistanceGreaterThan = 3f`

The documentation for this class was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/Photon↵
Network/Views/[PhotonTransformViewPositionModel.cs](#)

8.54 PhotonTransformViewRotationControl Class Reference

Public Member Functions

- [PhotonTransformViewRotationControl](#) ([PhotonTransformViewRotationModel](#) model)
- Quaternion [GetRotation](#) (Quaternion currentRotation)
- void [OnPhotonSerializeView](#) (Quaternion currentRotation, [PhotonStream](#) stream, [PhotonMessageInfo](#) info)

8.54.1 Constructor & Destructor Documentation

8.54.1.1 [PhotonTransformViewRotationControl.PhotonTransformViewRotationControl](#) ([PhotonTransformViewRotationModel](#) *model*)

8.54.2 Member Function Documentation

8.54.2.1 Quaternion [PhotonTransformViewRotationControl.GetRotation](#) (Quaternion *currentRotation*)

8.54.2.2 void [PhotonTransformViewRotationControl.OnPhotonSerializeView](#) (Quaternion *currentRotation*, [PhotonStream](#) *stream*, [PhotonMessageInfo](#) *info*)

The documentation for this class was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/PhotonNetwork/Views/[PhotonTransformViewRotationControl.cs](#)

8.55 PhotonTransformViewRotationModel Class Reference

Public Types

- enum [InterpolateOptions](#) { [InterpolateOptions.Disabled](#), [InterpolateOptions.RotateTowards](#), [InterpolateOptions.Lerp](#) }

Public Attributes

- bool [SynchronizeEnabled](#)
- [InterpolateOptions](#) [InterpolateOption](#) = [InterpolateOptions.RotateTowards](#)
- float [InterpolateRotateTowardsSpeed](#) = 180
- float [InterpolateLerpSpeed](#) = 5

8.55.1 Member Enumeration Documentation

8.55.1.1 enum [PhotonTransformViewRotationModel.InterpolateOptions](#) [strong]

Enumerator

Disabled

RotateTowards

Lerp

8.55.2 Member Data Documentation

8.55.2.1 float PhotonTransformViewRotationModel.InterpolateLerpSpeed = 5

8.55.2.2 InterpolateOptions PhotonTransformViewRotationModel.InterpolateOption = InterpolateOptions.Rotate↔
Towards

8.55.2.3 float PhotonTransformViewRotationModel.InterpolateRotateTowardsSpeed = 180

8.55.2.4 bool PhotonTransformViewRotationModel.SynchronizeEnabled

The documentation for this class was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/Photon↔
Network/Views/[PhotonTransformViewRotationModel.cs](#)

8.56 PhotonTransformViewScaleControl Class Reference

Public Member Functions

- [PhotonTransformViewScaleControl](#) ([PhotonTransformViewScaleModel](#) model)
- Vector3 [GetScale](#) (Vector3 currentScale)
- void [OnPhotonSerializeView](#) (Vector3 currentScale, [PhotonStream](#) stream, [PhotonMessageInfo](#) info)

8.56.1 Constructor & Destructor Documentation

8.56.1.1 PhotonTransformViewScaleControl.PhotonTransformViewScaleControl ([PhotonTransformViewScaleModel](#)
model)

8.56.2 Member Function Documentation

8.56.2.1 Vector3 PhotonTransformViewScaleControl.GetScale (Vector3 *currentScale*)

8.56.2.2 void PhotonTransformViewScaleControl.OnPhotonSerializeView (Vector3 *currentScale*, [PhotonStream](#) *stream*,
[PhotonMessageInfo](#) *info*)

The documentation for this class was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/Photon↔
Network/Views/[PhotonTransformViewScaleControl.cs](#)

8.57 PhotonTransformViewScaleModel Class Reference

Public Types

- enum [InterpolateOptions](#) { [InterpolateOptions.Disabled](#), [InterpolateOptions.MoveTowards](#), [Interpolate↔
Options.Lerp](#) }

Public Attributes

- bool [SynchronizeEnabled](#)
- [InterpolateOptions](#) [InterpolateOption](#) = [InterpolateOptions.Disabled](#)
- float [InterpolateMoveTowardsSpeed](#) = 1f
- float [InterpolateLerpSpeed](#)

8.57.1 Member Enumeration Documentation

8.57.1.1 enum [PhotonTransformViewScaleModel.InterpolateOptions](#) [strong]

Enumerator

Disabled

MoveTowards

Lerp

8.57.2 Member Data Documentation

8.57.2.1 float [PhotonTransformViewScaleModel.InterpolateLerpSpeed](#)

8.57.2.2 float [PhotonTransformViewScaleModel.InterpolateMoveTowardsSpeed](#) = 1f

8.57.2.3 [InterpolateOptions](#) [PhotonTransformViewScaleModel.InterpolateOption](#) = [InterpolateOptions.Disabled](#)

8.57.2.4 bool [PhotonTransformViewScaleModel.SynchronizeEnabled](#)

The documentation for this class was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/Photon↔
Network/Views/[PhotonTransformViewScaleModel.cs](#)

8.58 PhotonView Class Reference

PUN's [NetworkView](#) replacement class for networking.

Inherits [Photon.MonoBehaviour](#).

Public Member Functions

- void [RequestOwnership](#) ()

Depending on the [PhotonView](#)'s ownershipTransfer setting, any client can request to become owner of the [PhotonView](#).
- void [TransferOwnership](#) ([PhotonPlayer](#) newOwner)

Transfers the ownership of this [PhotonView](#) (and GameObject) to another player.
- void [TransferOwnership](#) (int newOwnerId)

Transfers the ownership of this [PhotonView](#) (and GameObject) to another player.
- void [SerializeView](#) ([PhotonStream](#) stream, [PhotonMessageInfo](#) info)
- void [DeserializeView](#) ([PhotonStream](#) stream, [PhotonMessageInfo](#) info)
- void [RefreshRpcMonoBehaviourCache](#) ()

Can be used to refresh the list of MonoBehaviours on this GameObject while [PhotonNetwork.UseRpcMonoBehaviourCache](#) is true.
- void [RPC](#) (string methodName, [PhotonTargets](#) target, params object[] parameters)

Call a RPC method of this GameObject on remote clients of this room (or on all, including this client).
- void [RpcSecure](#) (string methodName, [PhotonTargets](#) target, bool encrypt, params object[] parameters)

Call a RPC method of this GameObject on remote clients of this room (or on all, including this client).
- void [RPC](#) (string methodName, [PhotonPlayer](#) targetPlayer, params object[] parameters)

Call a RPC method of this GameObject on remote clients of this room (or on all, including this client).
- void [RpcSecure](#) (string methodName, [PhotonPlayer](#) targetPlayer, bool encrypt, params object[] parameters)

Call a RPC method of this GameObject on remote clients of this room (or on all, including this client).
- override string [ToString](#) ()

Static Public Member Functions

- static [PhotonView Get](#) (Component component)
- static [PhotonView Get](#) (GameObject gameObj)
- static [PhotonView Find](#) (int viewID)

Public Attributes

- int [ownerId](#)
- int [group](#) = 0
- int [prefixBackup](#) = -1
- Component [observed](#)
- [ViewSynchronization](#) synchronization
- [OnSerializeTransform](#) onSerializeTransformOption = [OnSerializeTransform.PositionAndRotation](#)
- [OnSerializeRigidBody](#) onSerializeRigidBodyOption = [OnSerializeRigidBody.All](#)
- [OwnershipOption](#) ownershipTransfer = [OwnershipOption.Fixed](#)

Defines if ownership of this [PhotonView](#) is fixed, can be requested or simply taken.
- List< Component > [ObservedComponents](#)
- int [instantiationId](#)

Properties

- int [prefix](#) [get, set]
- object[] [instantiationData](#) [get, set]

This is the instantiationData that was passed when calling [PhotonNetwork.Instantiate](#) (if that was used to spawn this prefab)*
- int [viewID](#) [get, set]

The ID of the [PhotonView](#).
- bool [isSceneView](#) [get]

True if the [PhotonView](#) was loaded with the scene (game object) or instantiated with [InstantiateSceneObject](#).
- [PhotonPlayer](#) owner [get]

The owner of a [PhotonView](#) is the player who created the GameObject with that view.
- int [OwnerActorNr](#) [get]
- bool [isOwnerActive](#) [get]
- int [CreatorActorNr](#) [get]
- bool [isMine](#) [get]

True if the [PhotonView](#) is "mine" and can be controlled by this client.

8.58.1 Detailed Description

PUN's NetworkView replacement class for networking.

Use it like a NetworkView.

8.58.2 Member Function Documentation

8.58.2.1 void [PhotonView.DeserializeView](#) ([PhotonStream](#) *stream*, [PhotonMessageInfo](#) *info*)

8.58.2.2 static [PhotonView](#) [PhotonView.Find](#) (int *viewID*) [static]

8.58.2.3 static [PhotonView](#) [PhotonView.Get](#) ([Component](#) *component*) [static]

8.58.2.4 static [PhotonView](#) [PhotonView.Get](#) ([GameObject](#) *gameObj*) [static]

8.58.2.5 void [PhotonView.RefreshRpcMonoBehaviourCache](#) ()

Can be used to refresh the list of MonoBehaviours on this GameObject while [PhotonNetwork.UseRpcMonoBehaviourCache](#) is true.

Set [PhotonNetwork.UseRpcMonoBehaviourCache](#) to true to enable the caching. Uses this.[GetComponent<MonoBehaviour>\(\)](#) to get a list of MonoBehaviours to call RPCs on (potentially).

While [PhotonNetwork.UseRpcMonoBehaviourCache](#) is false, this method has no effect, because the list is refreshed when a RPC gets called.

8.58.2.6 void PhotonView.RequestOwnership ()

Depending on the [PhotonView](#)'s ownershipTransfer setting, any client can request to become owner of the [PhotonView](#).

Requesting ownership can give you control over a [PhotonView](#), if the ownershipTransfer setting allows that. The current owner might have to implement [IPunCallbacks.OnOwnershipRequest](#) to react to the ownership request.

The owner/controller of a [PhotonView](#) is also the client which sends position updates of the GameObject.

8.58.2.7 void PhotonView.RPC (string methodName, PhotonTargets target, params object[] parameters)

Call a RPC method of this GameObject on remote clients of this room (or on all, including this client).

Remote Procedure Calls are an essential tool in making multiplayer games with PUN. It enables you to make every client in a room call a specific method.

RPC calls can target "All" or the "Others". Usually, the target "All" gets executed locally immediately after sending the RPC. The "*ViaServer" options send the RPC to the server and execute it on this client when it's sent back. Of course, calls are affected by this client's lag and that of remote clients.

Each call automatically is routed to the same [PhotonView](#) (and GameObject) that was used on the originating client.

See: Remote Procedure Calls.

Parameters

<i>methodName</i>	The name of a fitting method that was has the RPC attribute.
<i>target</i>	The group of targets and the way the RPC gets sent.
<i>parameters</i>	The parameters that the RPC method has (must fit this call!).

8.58.2.8 void PhotonView.RPC (string methodName, PhotonPlayer targetPlayer, params object[] parameters)

Call a RPC method of this GameObject on remote clients of this room (or on all, including this client).

Remote Procedure Calls are an essential tool in making multiplayer games with PUN. It enables you to make every client in a room call a specific method.

This method allows you to make an RPC calls on a specific player's client. Of course, calls are affected by this client's lag and that of remote clients.

Each call automatically is routed to the same [PhotonView](#) (and GameObject) that was used on the originating client.

See: Remote Procedure Calls.

Parameters

<i>methodName</i>	The name of a fitting method that was has the RPC attribute.
<i>targetPlayer</i>	The group of targets and the way the RPC gets sent.
<i>parameters</i>	The parameters that the RPC method has (must fit this call!).

8.58.2.9 void PhotonView.RpcSecure (string *methodName*, PhotonTargets *target*, bool *encrypt*, params object[] *parameters*)

Call a RPC method of this GameObject on remote clients of this room (or on all, including this client).

Remote Procedure Calls are an essential tool in making multiplayer games with PUN. It enables you to make every client in a room call a specific method.

RPC calls can target "All" or the "Others". Usually, the target "All" gets executed locally immediately after sending the RPC. The "*ViaServer" options send the RPC to the server and execute it on this client when it's sent back. Of course, calls are affected by this client's lag and that of remote clients.

Each call automatically is routed to the same PhotonView (and GameObject) that was used on the originating client.

See: Remote Procedure Calls.

param name="methodName">The name of a fitting method that has the RPC attribute.

param name="target">The group of targets and the way the RPC gets sent.

param name="encrypt">

param name="parameters">The parameters that the RPC method has (must fit this call!).

8.58.2.10 void PhotonView.RpcSecure (string *methodName*, PhotonPlayer *targetPlayer*, bool *encrypt*, params object[] *parameters*)

Call a RPC method of this GameObject on remote clients of this room (or on all, including this client).

Remote Procedure Calls are an essential tool in making multiplayer games with PUN. It enables you to make every client in a room call a specific method.

This method allows you to make an RPC calls on a specific player's client. Of course, calls are affected by this client's lag and that of remote clients.

Each call automatically is routed to the same PhotonView (and GameObject) that was used on the originating client.

See: Remote Procedure Calls.

param name="methodName">The name of a fitting method that has the RPC attribute.

param name="targetPlayer">The group of targets and the way the RPC gets sent.

param name="encrypt">

param name="parameters">The parameters that the RPC method has (must fit this call!).

8.58.2.11 void PhotonView.SerializeView (PhotonStream *stream*, PhotonMessageInfo *info*)

8.58.2.12 override string PhotonView.ToString ()

8.58.2.13 void PhotonView.TransferOwnership (PhotonPlayer *newOwner*)

Transfers the ownership of this PhotonView (and GameObject) to another player.

The owner/controller of a PhotonView is also the client which sends position updates of the GameObject.

8.58.2.14 void PhotonView.TransferOwnership (int *newOwnerId*)

Transfers the ownership of this [PhotonView](#) (and GameObject) to another player.

The owner/controller of a [PhotonView](#) is also the client which sends position updates of the GameObject.

8.58.3 Member Data Documentation

8.58.3.1 int PhotonView.group = 0

8.58.3.2 int PhotonView.instantiationId

8.58.3.3 Component PhotonView.observed

8.58.3.4 List<Component> PhotonView.ObservedComponents

8.58.3.5 OnSerializeRigidBody PhotonView.onSerializeRigidBodyOption = OnSerializeRigidBody.All

8.58.3.6 OnSerializeTransform PhotonView.onSerializeTransformOption = OnSerializeTransform.PositionAndRotation↔

8.58.3.7 int PhotonView.ownerId

8.58.3.8 OwnershipOption PhotonView.ownershipTransfer = OwnershipOption.Fixed

Defines if ownership of this [PhotonView](#) is fixed, can be requested or simply taken.

Note that you can't edit this value at runtime. The options are described in enum OwnershipOption. The current owner has to implement [IPunCallbacks.OnOwnershipRequest](#) to react to the ownership request.

8.58.3.9 int PhotonView.prefixBackup = -1

8.58.3.10 ViewSynchronization PhotonView.synchronization

8.58.4 Property Documentation

8.58.4.1 int PhotonView.CreatorActorNr [get]

8.58.4.2 object [] PhotonView.instantiationData [get], [set]

This is the instantiationData that was passed when calling [PhotonNetwork.Instantiate*](#) (if that was used to spawn this prefab)

8.58.4.3 bool PhotonView.isMine [get]

True if the [PhotonView](#) is "mine" and can be controlled by this client.

PUN has an ownership concept that defines who can control and destroy each [PhotonView](#). True in case the owner matches the local [PhotonPlayer](#). True if this is a scene photonview on the Master client.

8.58.4.4 bool PhotonView.isOwnerActive [get]**8.58.4.5 bool PhotonView.isSceneView** [get]

True if the [PhotonView](#) was loaded with the scene (game object) or instantiated with `InstantiateSceneObject`.

Scene objects are not owned by a particular player but belong to the scene. Thus they don't get destroyed when their creator leaves the game and the current Master Client can control them (whoever that is). The ownerId is 0 (player IDs are 1 and up).

8.58.4.6 PhotonPlayer PhotonView.owner [get]

The owner of a [PhotonView](#) is the player who created the `GameObject` with that view.

Objects in the scene don't have an owner.

The owner/controller of a [PhotonView](#) is also the client which sends position updates of the `GameObject`.

Ownership can be transferred to another player with [PhotonView.TransferOwnership](#) or any player can request ownership by calling the [PhotonView](#)'s `RequestOwnership` method. The current owner has to implement [IPunCallbacks.OnOwnershipRequest](#) to react to the ownership request.

8.58.4.7 int PhotonView.OwnerActorNr [get]**8.58.4.8 int PhotonView.prefix** [get], [set]**8.58.4.9 int PhotonView.viewID** [get], [set]

The ID of the [PhotonView](#).

Identifies it in a networked game (per room).

See: Network Instantiation

The documentation for this class was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/PhotonNetwork/[PhotonView.cs](#)

8.59 PhotonViewHandler Class Reference

Inherits `EditorWindow`.

Static Public Member Functions

- static int [GetID](#) (int idOffset, HashSet< int > usedInstanceViewNumbers)
- static void [LoadAllScenesToFix](#) ()

8.59.1 Member Function Documentation

8.59.1.1 static int PhotonViewHandler.GetID (int *idOffset*, HashSet< int > *usedInstanceViewNumbers*) [static]

8.59.1.2 static void PhotonViewHandler.LoadAllScenesToFix () [static]

The documentation for this class was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Editor/Photon↔ Network/[PhotonViewHandler.cs](#)

8.60 PhotonViewInspector Class Reference

Inherits Editor.

Public Member Functions

- override void [OnInspectorGUI](#) ()

8.60.1 Member Function Documentation

8.60.1.1 override void PhotonViewInspector.OnInspectorGUI ()

The documentation for this class was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Editor/Photon↔ Network/[PhotonViewInspector.cs](#)

8.61 PickupItem Class Reference

Makes a scene object pickup-able.

Inherits [Photon.MonoBehaviour](#), and [IPunObservable](#).

Public Member Functions

- void [OnTriggerEnter](#) (Collider other)
- void [OnPhotonSerializeView](#) ([PhotonStream](#) stream, [PhotonMessageInfo](#) info)
Called by PUN several times per second, so that your script can write and read synchronization data for the [PhotonView](#).
- void [Pickup](#) ()
- void [Drop](#) ()
Makes use of RPC PunRespawn to drop an item (sent through server for all).
- void [Drop](#) (Vector3 newPosition)
Makes use of RPC PunRespawn to drop an item (sent through server for all).
- void [PunPickup](#) ([PhotonMessageInfo](#) msgInfo)

Public Attributes

- float [SecondsBeforeRespawn](#) = 2
Enables you to define a timeout when the picked up item should re-spawn at the same place it was before.
- bool [PickupOnTrigger](#)
The most likely trigger to pick up an item.
- bool [PickupIsMine](#)
If the pickup item is currently yours. Interesting in OnPickedUp(PickupItem item).
- MonoBehaviour [OnPickedUpCall](#)
GameObject to send an event "OnPickedUp(PickupItem item)" to.
- bool [SentPickup](#)
If this client sent a pickup. To avoid sending multiple pickup requests before reply is there.
- double [TimeOfRespawn](#)
Timestamp when to respawn the item (compared to [PhotonNetwork.time](#)).

Static Public Attributes

- static HashSet< [PickupItem](#) > [DisabledPickupItems](#) = new HashSet<[PickupItem](#)>()

Properties

- int [ViewID](#) [get]

8.61.1 Detailed Description

Makes a scene object pickup-able.

Needs a [PhotonView](#) which belongs to the scene.

Includes a OnPhotonSerializeView implementation that

8.61.2 Member Function Documentation

8.61.2.1 void PickupItem.Drop ()

Makes use of RPC PunRespawn to drop an item (sent through server for all).

8.61.2.2 void PickupItem.Drop (Vector3 newPosition)

Makes use of RPC PunRespawn to drop an item (sent through server for all).

8.61.2.3 void PickupItem.OnPhotonSerializeView (PhotonStream stream, PhotonMessageInfo info)

Called by PUN several times per second, so that your script can write and read synchronization data for the [PhotonView](#).

This method will be called in scripts that are assigned as Observed component of a [PhotonView](#).

[PhotonNetwork.sendRateOnSerialize](#) affects how often this method is called.

[PhotonNetwork.sendRate](#) affects how often packages are sent by this client.

Implementing this method, you can customize which data a [PhotonView](#) regularly synchronizes. Your code defines what is being sent (content) and how your data is used by receiving clients.

Unlike other callbacks, *OnPhotonSerializeView* only gets called when it is assigned to a [PhotonView](#) as [PhotonView.observed](#) script.

To make use of this method, the [PhotonStream](#) is essential. It will be in "writing" mode" on the client that controls a PhotonView ([PhotonStream.isWriting](#) == true) and in "reading mode" on the remote clients that just receive that the controlling client sends.

If you skip writing any value into the stream, PUN will skip the update. Used carefully, this can conserve bandwidth and messages (which have a limit per room/second).

Note that *OnPhotonSerializeView* is not called on remote clients when the sender does not send any update. This can't be used as "x-times per second Update()".

Implements [IPunObservable](#).

8.61.2.4 void PickupItem.OnTriggerEnter (Collider other)

8.61.2.5 void PickupItem.Pickup ()

8.61.2.6 void PickupItem.PunPickup (PhotonMessageInfo msgInfo)

8.61.3 Member Data Documentation

8.61.3.1 HashSet<PickupItem> PickupItem.DisabledPickupItems = new HashSet<PickupItem>() [static]

8.61.3.2 MonoBehaviour PickupItem.OnPickedUpCall

GameObject to send an event "OnPickedUp(PickupItem item)" to.

Implement *OnPickedUp(PickupItem item) {}* in some script on the linked game object. The item will be "this" and *item.PickupsMine* will help you to find if this pickup was done by "this player".

8.61.3.3 bool PickupItem.PickupIsMine

If the pickup item is currently yours. Interesting in OnPickedUp(PickupItem item).

8.61.3.4 bool PickupItem.PickupOnTrigger

The most likely trigger to pick up an item.

Set in inspector!

Edit the collider and set collision masks to avoid pickups by random objects.

8.61.3.5 float PickupItem.SecondsBeforeRespawn = 2

Enables you to define a timeout when the picked up item should re-spawn at the same place it was before.

Set in Inspector per GameObject! The value in code is just the default.

If you don't want an item to respawn, set SecondsBeforeRespawn == 0. If an item does not respawn, it could be consumed or carried around and dropped somewhere else.

A respawning item should stick to a fixed position. It should not be observed at all (in any [PhotonView](#)). It can only be consumed and can't be dropped somewhere else (cause that would double the item).

This script uses PunRespawn() as RPC and as method that gets called by Invoke() after a timeout. No matter if the item respawns timed or by Drop, that method makes sure (temporary) owner and other status-values are being re-set.

8.61.3.6 bool PickupItem.SentPickup

If this client sent a pickup. To avoid sending multiple pickup requests before reply is there.

8.61.3.7 double PickupItem.TimeOfRespawn

Timestamp when to respawn the item (compared to [PhotonNetwork.time](#)).

8.61.4 Property Documentation

8.61.4.1 int PickupItem.ViewID [get]

The documentation for this class was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/UtilityScripts/[PickupItem.cs](#)

8.62 PickupItemSimple Class Reference

Makes a scene object pickup-able.

Inherits [Photon.MonoBehaviour](#).

Public Member Functions

- void [OnTriggerEnter](#) (Collider other)
- void [Pickup](#) ()
- void [PunPickupSimple](#) ([PhotonMessageInfo](#) msgInfo)
- void [RespawnAfter](#) ()

Public Attributes

- float [SecondsBeforeRespawn](#) = 2
- bool [PickupOnCollide](#)
- bool [SentPickup](#)

Additional Inherited Members

8.62.1 Detailed Description

Makes a scene object pickup-able.

Needs a [PhotonView](#) which belongs to the scene.

8.62.2 Member Function Documentation

8.62.2.1 void PickupItemSimple.OnTriggerEnter (Collider *other*)

8.62.2.2 void PickupItemSimple.Pickup ()

8.62.2.3 void PickupItemSimple.PunPickupSimple (PhotonMessageInfo *msgInfo*)

8.62.2.4 void PickupItemSimple.RespawnAfter ()

8.62.3 Member Data Documentation

8.62.3.1 bool PickupItemSimple.PickupOnCollide

8.62.3.2 float PickupItemSimple.SecondsBeforeRespawn = 2

8.62.3.3 bool PickupItemSimple.SentPickup

The documentation for this class was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/UtilityScripts/[PickupItemSimple.cs](#)

8.63 PickupItemSyncer Class Reference

Finds out which PickupItems are not spawned at the moment and send this to new players.

Inherits [Photon.MonoBehaviour](#).

Public Member Functions

- void [OnPhotonPlayerConnected](#) ([PhotonPlayer](#) newPlayer)
- void [OnJoinedRoom](#) ()
- void [AskForPickupItemSpawnTimes](#) ()
- void [RequestForPickupTimes](#) ([PhotonMessageInfo](#) msgInfo)
- void [PickupItemInit](#) (double timeBase, float[] inactivePickupsAndTimes)

Public Attributes

- bool [IsWaitingForPickupInit](#)

Additional Inherited Members

8.63.1 Detailed Description

Finds out which PickupItems are not spawned at the moment and send this to new players.

Attach this component to a single GameObject in the scene, not to all PickupItems.

8.63.2 Member Function Documentation

8.63.2.1 void PickupItemSyncer.AskForPickupItemSpawnTimes ()

8.63.2.2 void PickupItemSyncer.OnJoinedRoom ()

8.63.2.3 void PickupItemSyncer.OnPhotonPlayerConnected ([PhotonPlayer](#) newPlayer)

8.63.2.4 void PickupItemSyncer.PickupItemInit (double timeBase, float[] inactivePickupsAndTimes)

8.63.2.5 void PickupItemSyncer.RequestForPickupTimes ([PhotonMessageInfo](#) msgInfo)

8.63.3 Member Data Documentation

8.63.3.1 bool PickupItemSyncer.IsWaitingForPickupInit

The documentation for this class was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/UtilityScripts/[PickupItemSyncer.cs](#)

8.64 PingMonoEditor Class Reference

Uses C# Socket class from System.Net.Sockets (as Unity usually does).

Inherits PhotonPing.

Public Member Functions

- override bool [StartPing](#) (string ip)
- override bool [Done](#) ()
- override void [Dispose](#) ()

8.64.1 Detailed Description

Uses C# Socket class from System.Net.Sockets (as Unity usually does).

Incompatible with Windows 8 Store/Phone API.

8.64.2 Member Function Documentation

8.64.2.1 override void PingMonoEditor.Dispose ()

8.64.2.2 override bool PingMonoEditor.Done ()

8.64.2.3 override bool PingMonoEditor.StartPing (string ip)

The documentation for this class was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/Photon↔ Network/[PingCloudRegions.cs](#)

8.65 PointedAtGameObjectInfo Class Reference

Inherits MonoBehaviour.

The documentation for this class was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/UtilityScripts/[Pointed↔ AtGameObjectInfo.cs](#)

8.66 Photon.PunBehaviour Class Reference

This class provides a .photonView and all callbacks/events that PUN can call.

Inherits [Photon.MonoBehaviour](#), and [IPunCallbacks](#).

Public Member Functions

- virtual void [OnConnectedToPhoton](#) ()
Called when the initial connection got established but before you can use the server.
- virtual void [OnLeftRoom](#) ()
Called when the local user/client left a room.
- virtual void [OnMasterClientSwitched](#) ([PhotonPlayer](#) newMasterClient)
Called after switching to a new MasterClient when the current one leaves.
- virtual void [OnPhotonCreateRoomFailed](#) (object[] codeAndMsg)
Called when a CreateRoom() call failed.
- virtual void [OnPhotonJoinRoomFailed](#) (object[] codeAndMsg)
Called when a JoinRoom() call failed.
- virtual void [OnCreatedRoom](#) ()
Called when this client created a room and entered it.
- virtual void [OnJoinedLobby](#) ()
Called on entering a lobby on the Master Server.
- virtual void [OnLeftLobby](#) ()
Called after leaving a lobby.
- virtual void [OnFailedToConnectToPhoton](#) ([DisconnectCause](#) cause)
Called if a connect call to the [Photon](#) server failed before the connection was established, followed by a call to [OnDisconnectedFromPhoton\(\)](#).
- virtual void [OnDisconnectedFromPhoton](#) ()
Called after disconnecting from the [Photon](#) server.
- virtual void [OnConnectionFail](#) ([DisconnectCause](#) cause)
Called when something causes the connection to fail (after it was established), followed by a call to [OnDisconnectedFromPhoton\(\)](#).
- virtual void [OnPhotonInstantiate](#) ([PhotonMessageInfo](#) info)
Called on all scripts on a GameObject (and children) that have been Instantiated using [PhotonNetwork.Instantiate](#).
- virtual void [OnReceivedRoomListUpdate](#) ()
Called for any update of the room-listing while in a lobby ([PhotonNetwork.insideLobby](#)) on the Master Server.
- virtual void [OnJoinedRoom](#) ()
Called when entering a room (by creating or joining it).
- virtual void [OnPhotonPlayerConnected](#) ([PhotonPlayer](#) newPlayer)
Called when a remote player entered the room.
- virtual void [OnPhotonPlayerDisconnected](#) ([PhotonPlayer](#) otherPlayer)
Called when a remote player left the room.
- virtual void [OnPhotonRandomJoinFailed](#) (object[] codeAndMsg)
Called when a JoinRandom() call failed.
- virtual void [OnConnectedToMaster](#) ()
Called after the connection to the master is established and authenticated but only when [PhotonNetwork.autoJoinLobby](#) is false.
- virtual void [OnPhotonMaxCccuReached](#) ()
Because the concurrent user limit was (temporarily) reached, this client is rejected by the server and disconnecting.
- virtual void [OnPhotonCustomRoomPropertiesChanged](#) ([Hashtable](#) propertiesThatChanged)
Called when a room's custom properties changed.
- virtual void [OnPhotonPlayerPropertiesChanged](#) (object[] playerAndUpdatedProps)
Called when custom player-properties are changed.
- virtual void [OnUpdatedFriendList](#) ()
Called when the server sent the response to a FindFriends request and updated [PhotonNetwork.Friends](#).
- virtual void [OnCustomAuthenticationFailed](#) (string debugMessage)
Called when the custom authentication failed.
- virtual void [OnWebRpcResponse](#) ([OperationResponse](#) response)

Called by PUN when the response to a WebRPC is available.

- virtual void [OnOwnershipRequest](#) (object[] viewAndPlayer)

Called when another player requests ownership of a [PhotonView](#) from you (the current owner).

- virtual void [OnLobbyStatisticsUpdate](#) ()

Called when the Master Server sent an update for the Lobby Statistics, updating [PhotonNetwork.LobbyStatistics](#).

Additional Inherited Members

8.66.1 Detailed Description

This class provides a `.photonView` and all callbacks/events that PUN can call.

Override the events/methods you want to use.

By extending this class, you can implement individual methods as override.

Visual Studio and MonoDevelop should provide the list of methods when you begin typing "override". **Your implementation does not have to call "base.method()".**

This class implements [IPunCallbacks](#), which is used as definition of all PUN callbacks. Don't implement [IPunCallbacks](#) in your classes. Instead, implement [PunBehaviour](#) or individual methods.

8.66.2 Member Function Documentation

8.66.2.1 virtual void Photon.PunBehaviour.OnConnectedToMaster () [virtual]

Called after the connection to the master is established and authenticated but only when [PhotonNetwork.autoJoinLobby](#) is false.

If you set [PhotonNetwork.autoJoinLobby](#) to true, [OnJoinedLobby\(\)](#) will be called instead of this.

You can join rooms and create them even without being in a lobby. The default lobby is used in that case. The list of available rooms won't become available unless you join a lobby via [PhotonNetwork.joinLobby](#).

Implements [IPunCallbacks](#).

8.66.2.2 virtual void Photon.PunBehaviour.OnConnectedToPhoton () [virtual]

Called when the initial connection got established but before you can use the server.

[OnJoinedLobby\(\)](#) or [OnConnectedToMaster\(\)](#) are called when PUN is ready.

This callback is only useful to detect if the server can be reached at all (technically). Most often, it's enough to implement [OnFailedToConnectToPhoton\(\)](#) and [OnDisconnectedFromPhoton\(\)](#).

[OnJoinedLobby\(\)](#) or [OnConnectedToMaster\(\)](#) are called when PUN is ready.

When this is called, the low level connection is established and PUN will send your AppId, the user, etc in the background. This is not called for transitions from the masterserver to game servers.

Implements [IPunCallbacks](#).

8.66.2.3 virtual void Photon.PunBehaviour.OnConnectionFail (DisconnectCause *cause*) [virtual]

Called when something causes the connection to fail (after it was established), followed by a call to [OnDisconnectedFromPhoton\(\)](#).

If the server could not be reached in the first place, `OnFailedToConnectToPhoton` is called instead. The reason for the error is provided as `DisconnectCause`.

Implements [IPunCallbacks](#).

8.66.2.4 virtual void Photon.PunBehaviour.OnCreatedRoom () [virtual]

Called when this client created a room and entered it.

[OnJoinedRoom\(\)](#) will be called as well.

This callback is only called on the client which created a room (see [PhotonNetwork.CreateRoom\(\)](#)).

As any client might close (or drop connection) anytime, there is a chance that the creator of a room does not execute `OnCreatedRoom`.

If you need specific room properties or a "start signal", it is safer to implement [OnMasterClientSwitched\(\)](#) and to make the new `MasterClient` check the room's state.

Implements [IPunCallbacks](#).

8.66.2.5 virtual void Photon.PunBehaviour.OnCustomAuthenticationFailed (string *debugMessage*) [virtual]

Called when the custom authentication failed.

Followed by disconnect!

Custom Authentication can fail due to user-input, bad tokens/secrets. If authentication is successful, this method is not called. Implement [OnJoinedLobby\(\)](#) or [OnConnectedToMaster\(\)](#) (as usual).

During development of a game, it might also fail due to wrong configuration on the server side. In those cases, logging the `debugMessage` is very important.

Unless you setup a custom authentication service for your app (in the [Dashboard](#)), this won't be called!

Parameters

<i>debugMessage</i>	Contains a debug message why authentication failed. This has to be fixed during development time.
---------------------	---

Implements [IPunCallbacks](#).

8.66.2.6 virtual void Photon.PunBehaviour.OnDisconnectedFromPhoton () [virtual]

Called after disconnecting from the [Photon](#) server.

In some cases, other callbacks are called before `OnDisconnectedFromPhoton` is called. Examples: [OnConnectionFail\(\)](#) and [OnFailedToConnectToPhoton\(\)](#).

Implements [IPunCallbacks](#).

8.66.2.7 virtual void Photon.PunBehaviour.OnFailedToConnectToPhoton (DisconnectCause *cause*) [virtual]

Called if a connect call to the [Photon](#) server failed before the connection was established, followed by a call to [OnDisconnectedFromPhoton\(\)](#).

This is called when no connection could be established at all. It differs from `OnConnectionFail`, which is called when an existing connection fails.

Implements [IPunCallbacks](#).

8.66.2.8 virtual void Photon.PunBehaviour.OnJoinedLobby () [virtual]

Called on entering a lobby on the Master Server.

The actual room-list updates will call [OnReceivedRoomListUpdate\(\)](#).

Note: When [PhotonNetwork.autoJoinLobby](#) is false, [OnConnectedToMaster\(\)](#) will be called and the room list won't become available.

While in the lobby, the roomlist is automatically updated in fixed intervals (which you can't modify). The room list gets available when [OnReceivedRoomListUpdate\(\)](#) gets called after [OnJoinedLobby\(\)](#).

Implements [IPunCallbacks](#).

8.66.2.9 virtual void Photon.PunBehaviour.OnJoinedRoom () [virtual]

Called when entering a room (by creating or joining it).

Called on all clients (including the Master Client).

This method is commonly used to instantiate player characters. If a match has to be started "actively", you can call an [PunRPC](#) triggered by a user's button-press or a timer.

When this is called, you can usually already access the existing players in the room via [PhotonNetwork.playerList](#). Also, all custom properties should be already available as [Room.customProperties](#). Check [Room.playerCount](#) to find out if enough players are in the room to start playing.

Implements [IPunCallbacks](#).

8.66.2.10 virtual void Photon.PunBehaviour.OnLeftLobby () [virtual]

Called after leaving a lobby.

When you leave a lobby, [CreateRoom](#) and [JoinRandomRoom](#) automatically refer to the default lobby.

Implements [IPunCallbacks](#).

8.66.2.11 virtual void Photon.PunBehaviour.OnLeftRoom () [virtual]

Called when the local user/client left a room.

When leaving a room, PUN brings you back to the Master Server. Before you can use lobbies and join or create rooms, [OnJoinedLobby\(\)](#) or [OnConnectedToMaster\(\)](#) will get called again.

Implements [IPunCallbacks](#).

8.66.2.12 virtual void Photon.PunBehaviour.OnLobbyStatisticsUpdate () [virtual]

Called when the Master Server sent an update for the Lobby Statistics, updating [PhotonNetwork.LobbyStatistics](#).

This callback has two preconditions: EnableLobbyStatistics must be set to true, before this client connects. And the client has to be connected to the Master Server, which is providing the info about lobbies.

Implements [IPunCallbacks](#).

8.66.2.13 virtual void Photon.PunBehaviour.OnMasterClientSwitched (PhotonPlayer newMasterClient) [virtual]

Called after switching to a new MasterClient when the current one leaves.

This is not called when this client enters a room. The former MasterClient is still in the player list when this method get called.

Implements [IPunCallbacks](#).

8.66.2.14 virtual void Photon.PunBehaviour.OnOwnershipRequest (object[] viewAndPlayer) [virtual]

Called when another player requests ownership of a [PhotonView](#) from you (the current owner).

The parameter viewAndPlayer contains:

[PhotonView](#) view = viewAndPlayer[0] as [PhotonView](#);

[PhotonPlayer](#) requestingPlayer = viewAndPlayer[1] as [PhotonPlayer](#);

Parameters

<i>viewAndPlayer</i>	The PhotonView is viewAndPlayer[0] and the requesting player is viewAndPlayer[1].
----------------------	---

Implements [IPunCallbacks](#).

8.66.2.15 virtual void Photon.PunBehaviour.OnPhotonCreateRoomFailed (object[] codeAndMsg) [virtual]

Called when a CreateRoom() call failed.

The parameter provides ErrorCode and message (as array).

Most likely because the room name is already in use (some other client was faster than you). PUN logs some info if the [PhotonNetwork.logLevel](#) is \geq [PhotonLogLevel.Informational](#).

Parameters

<i>codeAndMsg</i>	codeAndMsg[0] is a short ErrorCode and codeAndMsg[1] is a string debug msg.
-------------------	---

Implements [IPunCallbacks](#).

8.66.2.16 virtual void Photon.PunBehaviour.OnPhotonCustomRoomPropertiesChanged (Hashtable *propertiesThatChanged*)
[virtual]

Called when a room's custom properties changed.

The propertiesThatChanged contains all that was set via [Room.SetCustomProperties](#).

Since v1.25 this method has one parameter: Hashtable propertiesThatChanged.

Changing properties must be done by [Room.SetCustomProperties](#), which causes this callback locally, too.

Parameters

<i>propertiesThatChanged</i>	
------------------------------	--

Implements [IPunCallbacks](#).

8.66.2.17 virtual void Photon.PunBehaviour.OnPhotonInstantiate (PhotonMessageInfo *info*) [virtual]

Called on all scripts on a GameObject (and children) that have been Instantiated using [PhotonNetwork.Instantiate](#).

[PhotonMessageInfo](#) parameter provides info about who created the object and when (based off Photon↔ Networking.time).

Implements [IPunCallbacks](#).

8.66.2.18 virtual void Photon.PunBehaviour.OnPhotonJoinRoomFailed (object[] *codeAndMsg*) [virtual]

Called when a JoinRoom() call failed.

The parameter provides ErrorCode and message (as array).

Most likely error is that the room does not exist or the room is full (some other client was faster than you). PUN logs some info if the [PhotonNetwork.logLevel](#) is >= [PhotonLogLevel.Informational](#).

Parameters

<i>codeAndMsg</i>	codeAndMsg[0] is short ErrorCode. codeAndMsg[1] is string debug msg.
-------------------	--

Implements [IPunCallbacks](#).

8.66.2.19 `virtual void Photon.PunBehaviour.OnPhotonMaxCcuReached () [virtual]`

Because the concurrent user limit was (temporarily) reached, this client is rejected by the server and disconnecting.

When this happens, the user might try again later. You can't create or join rooms in `OnPhotonMaxCcuReached()`, cause the client will be disconnecting. You can raise the CCU limits with a new license (when you host yourself) or extended subscription (when using the [Photon](#) Cloud). The [Photon](#) Cloud will mail you when the CCU limit was reached. This is also visible in the Dashboard (webpage).

Implements [IPunCallbacks](#).

8.66.2.20 `virtual void Photon.PunBehaviour.OnPhotonPlayerConnected (PhotonPlayer newPlayer) [virtual]`

Called when a remote player entered the room.

This [PhotonPlayer](#) is already added to the playerlist at this time.

If your game starts with a certain number of players, this callback can be useful to check the [Room.playerCount](#) and find out if you can start.

Implements [IPunCallbacks](#).

8.66.2.21 `virtual void Photon.PunBehaviour.OnPhotonPlayerDisconnected (PhotonPlayer otherPlayer) [virtual]`

Called when a remote player left the room.

This [PhotonPlayer](#) is already removed from the playerlist at this time.

When your client calls `PhotonNetwork.leaveRoom`, PUN will call this method on the remaining clients. When a remote client drops connection or gets closed, this callback gets executed. after a timeout of several seconds.

Implements [IPunCallbacks](#).

8.66.2.22 `virtual void Photon.PunBehaviour.OnPhotonPlayerPropertiesChanged (object[] playerAndUpdatedProps) [virtual]`

Called when custom player-properties are changed.

Player and the changed properties are passed as `object[]`.

Since v1.25 this method has one parameter: `object[] playerAndUpdatedProps`, which contains two entries.

[0] is the affected [PhotonPlayer](#).

[1] is the Hashtable of properties that changed.

We are using a `object[]` due to limitations of Unity's `GameObject.SendMessage` (which has only one optional parameter).

Changing properties must be done by [PhotonPlayer.SetCustomProperties](#), which causes this callback locally, too.

Example:

```
void OnPhotonPlayerPropertiesChanged(object[] playerAndUpdatedProps) {
    PhotonPlayer player = playerAndUpdatedProps[0] as PhotonPlayer;
    Hashtable props = playerAndUpdatedProps[1] as Hashtable;
    //...
}
```

Parameters

<i>playerAndUpdatedProps</i>	Contains PhotonPlayer and the properties that changed See remarks.
------------------------------	--

Implements [IPunCallbacks](#).

8.66.2.23 `virtual void Photon.PunBehaviour.OnPhotonRandomJoinFailed (object[] codeAndMsg) [virtual]`

Called when a JoinRandom() call failed.

The parameter provides ErrorCode and message.

Most likely all rooms are full or no rooms are available.

When using multiple lobbies (via JoinLobby or [TypedLobby](#)), another lobby might have more/fitting rooms.

PUN logs some info if the [PhotonNetwork.logLevel](#) is \geq [PhotonLogLevel.Informational](#).

Parameters

<i>codeAndMsg</i>	codeAndMsg[0] is short ErrorCode. codeAndMsg[1] is string debug msg.
-------------------	--

Implements [IPunCallbacks](#).

8.66.2.24 `virtual void Photon.PunBehaviour.OnReceivedRoomListUpdate () [virtual]`

Called for any update of the room-listing while in a lobby ([PhotonNetwork.insideLobby](#)) on the Master Server.

PUN provides the list of rooms by [PhotonNetwork.GetRoomList\(\)](#).

Each item is a [RoomInfo](#) which might include custom properties (provided you defined those as lobby-listed when creating a room).

Not all types of lobbies provide a listing of rooms to the client. Some are silent and specialized for server-side matchmaking.

Implements [IPunCallbacks](#).

8.66.2.25 `virtual void Photon.PunBehaviour.OnUpdatedFriendList () [virtual]`

Called when the server sent the response to a FindFriends request and updated [PhotonNetwork.Friends](#).

The friends list is available as [PhotonNetwork.Friends](#), listing name, online state and the room a user is in (if any).

Implements [IPunCallbacks](#).

8.66.2.26 virtual void Photon.PunBehaviour.OnWebRpcResponse (*OperationResponse response*) [virtual]

Called by PUN when the response to a WebRPC is available.

See PhotonNetwork.WebRPC.

Important: The response.ReturnCode is 0 if [Photon](#) was able to reach your web-service. The content of the response is what your web-service sent. You can create a WebResponse instance from it. Example: [WebRpcResponse](#) webResponse = new [WebRpcResponse\(operationResponse\)](#);

Please note: Class OperationResponse is in a namespace which needs to be "used": using [ExitGames.Client.Photon](#); // includes OperationResponse (and other classes)

The OperationResponse.ReturnCode by [Photon](#) is:

```
0 for "OK"
-3 for "Web-Service not configured" (see Dashboard / WebHooks)
-5 for "Web-Service does now have RPC path/name" (at least for Azure)
```

Implements [IPunCallbacks](#).

The documentation for this class was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/PhotonNetwork/[PhotonClasses.cs](#)

8.67 PunPlayerScores Class Reference

Inherits MonoBehaviour.

Public Attributes

- const string [PlayerScoreProp](#) = "score"

8.67.1 Member Data Documentation

8.67.1.1 const string PunPlayerScores.PlayerScoreProp = "score"

The documentation for this class was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/UtilityScripts/[PunPlayerScores.cs](#)

8.68 PunRPC Class Reference

Replacement for RPC attribute with different name. Used to flag methods as remote-callable.

Inherits Attribute.

8.68.1 Detailed Description

Replacement for RPC attribute with different name. Used to flag methods as remote-callable.

The documentation for this class was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/Photon↔ Network/[RPC.cs](#)

8.69 PunSceneSettings Class Reference

Inherits `ScriptableObject`.

Static Public Member Functions

- static int [MinViewIdForScene](#) (string scene)

Public Attributes

- List< [SceneSetting](#) > [MinViewIdPerScene](#) = new List<[SceneSetting](#)>()

Properties

- static string [PunSceneSettingsCsPath](#) [get]
- static [PunSceneSettings Instance](#) [get]

8.69.1 Member Function Documentation

8.69.1.1 static int `PunSceneSettings.MinViewIdForScene (string scene)` [static]

8.69.2 Member Data Documentation

8.69.2.1 List<[SceneSetting](#)> `PunSceneSettings.MinViewIdPerScene = new List<SceneSetting>()`

8.69.3 Property Documentation

8.69.3.1 [PunSceneSettings PunSceneSettings.Instance](#) [static], [get]

8.69.3.2 string `PunSceneSettings.PunSceneSettingsCsPath` [static], [get]

The documentation for this class was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Editor/Photon↔ Network/[PunSceneSettings.cs](#)

8.70 PunTeams Class Reference

Implements teams in a room/game with help of player properties.

Inherits MonoBehaviour.

Public Types

- enum [Team](#) : byte { [Team.none](#), [Team.red](#), [Team.blue](#) }
Enum defining the teams available. First team should be neutral (it's the default value any field of this enum gets).

Public Member Functions

- void [Start](#) ()
- void [OnJoinedRoom](#) ()
Needed to update the team lists when joining a room.
- void [OnPhotonPlayerPropertiesChanged](#) (object[] playerAndUpdatedProps)
Refreshes the team lists.
- void [UpdateTeams](#) ()

Public Attributes

- const string [TeamPlayerProp](#) = "team"
Defines the player custom property name to use for team affinity of "this" player.

Static Public Attributes

- static Dictionary< [Team](#), List< [PhotonPlayer](#) > > [PlayersPerTeam](#)
The main list of teams with their player-lists.

8.70.1 Detailed Description

Implements teams in a room/game with help of player properties.

Access them by PhotonPlayer.GetTeam extension.

Teams are defined by enum Team. Change this to get more / different teams. There are no rules when / if you can join a team. You could add this in JoinTeam or something.

8.70.2 Member Enumeration Documentation

8.70.2.1 enum PunTeams.Team : byte [strong]

Enum defining the teams available. First team should be neutral (it's the default value any field of this enum gets).

Enumerator

none

red

blue

8.70.3 Member Function Documentation

8.70.3.1 void PunTeams.OnJoinedRoom ()

Needed to update the team lists when joining a room.

Called by PUN. See enum PhotonNetworkingMessage for an explanation.

8.70.3.2 void PunTeams.OnPhotonPlayerPropertiesChanged (object[] *playerAndUpdatedProps*)

Refreshes the team lists.

It could be a non-team related property change, too.

Called by PUN. See enum PhotonNetworkingMessage for an explanation.

8.70.3.3 void PunTeams.Start ()

8.70.3.4 void PunTeams.UpdateTeams ()

8.70.4 Member Data Documentation

8.70.4.1 Dictionary<Team, List<PhotonPlayer>> PunTeams.PlayersPerTeam [static]

The main list of teams with their player-lists.

Automatically kept up to date.

Note that this is static. Can be accessed by PunTeam.PlayersPerTeam. You should not modify this.

8.70.4.2 const string PunTeams.TeamPlayerProp = "team"

Defines the player custom property name to use for team affinity of "this" player.

The documentation for this class was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/UtilityScripts/[PunTeams.cs](#)

8.71 PunWizardText Class Reference

Public Attributes

- string [WindowTitle](#) = "PUN Wizard"
- string [SetupWizardWarningTitle](#) = "Warning"
- string [SetupWizardWarningMessage](#) = "You have not yet run the Photon setup wizard! Your game won't be able to connect. See Windows -> Photon Unity Networking."
- string [MainMenuButton](#) = "Main Menu"
- string [SetupWizardTitle](#) = "PUN Setup"
- string [SetupWizardInfo](#) = "Thanks for importing Photon Unity Networking.\nThis window should set you up.\n\n- To use an existing Photon Cloud App, enter your AppId.\n- To register an account or access an existing one, enter the account's mail address.\n- To use Photon OnPremise, skip this step."
- string [EmailOrAppIdLabel](#) = "AppId or Email"
- string [AlreadyRegisteredInfo](#) = "The email is registered so we can't fetch your AppId (without password).\n\n↔ Please login online to get your AppId and paste it above."
- string [SkipRegistrationInfo](#) = "Skipping? No problem:\nEdit your server settings in the PhotonServerSettings file."
- string [RegisteredNewAccountInfo](#) = "We created a (free) account and fetched you an AppId.\nWelcome. Your PUN project is setup."
- string [AppliedToSettingsInfo](#) = "Your AppId is now applied to this project."
- string [SetupCompleteInfo](#) = "Done!\nAll connection settings can be edited in the Photon↔ ServerSettings now.\nHave a look."
- string [CloseWindowButton](#) = "Close"
- string [SkipButton](#) = "Skip"
- string [SetupButton](#) = "Setup Project"
- string [MobileExportNoteLabel](#) = "Build for mobiles impossible. Get PUN+ or Unity Pro for mobile or use Unity 5."
- string [MobilePunPlusExportNoteLabel](#) = "PUN+ available. Using native sockets for iOS/Android."
- string [CancelButton](#) = "Cancel"
- string [PUNWizardLabel](#) = "PUN Wizard"
- string [SettingsButton](#) = "Settings"
- string [SetupServerCloudLabel](#) = "Setup wizard for setting up your own server or the cloud."
- string [WarningPhotonDisconnect](#) = ""
- string [ConverterLabel](#) = "Converter"
- string [StartButton](#) = "Start"
- string [UNtoPUNLabel](#) = "Converts pure Unity Networking to Photon Unity Networking."
- string [LocateSettingsButton](#) = "Locate PhotonServerSettings"
- string [SettingsHighlightLabel](#) = "Highlights the used photon settings file in the project."
- string [DocumentationLabel](#) = "Documentation"
- string [OpenPDFText](#) = "Reference PDF"
- string [OpenPDFTooltip](#) = "Opens the local documentation pdf."
- string [OpenDevNetText](#) = "DevNet / Manual"
- string [OpenDevNetTooltip](#) = "Online documentation for Photon."
- string [OpenCloudDashboardText](#) = "Cloud Dashboard Login"
- string [OpenCloudDashboardTooltip](#) = "Review Cloud App information and statistics."
- string [OpenForumText](#) = "Open Forum"
- string [OpenForumTooltip](#) = "Online support for Photon."
- string [OkButton](#) = "Ok"
- string [OwnHostCloudCompareLabel](#) = "I am not quite sure how 'my own host' compares to 'cloud'."
- string [ComparisonPageButton](#) = "Cloud versus OnPremise"
- string [ConnectionTitle](#) = "Connecting"
- string [ConnectionInfo](#) = "Connecting to the account service..."

- string `ErrorTextTitle` = "Error"
- string `IncorrectRPCListTitle` = "Warning: RPC-list becoming incompatible!"
- string `IncorrectRPCListLabel` = "Your project's RPC-list is full, so we can't add some RPCs just compiled.\n\nBy removing outdated RPCs, the list will be long enough but incompatible with older client builds!\n\nMake sure you change the game version where you use `PhotonNetwork.ConnectUsingSettings()`."
- string `RemoveOutdatedRPCsLabel` = "Remove outdated RPCs"
- string `FullRPCListTitle` = "Warning: RPC-list is full!"
- string `FullRPCListLabel` = "Your project's RPC-list is too long for PUN.\n\nYou can change PUN's source to use short-typed RPC index. Look for comments 'LIMITS RPC COUNT'\n\nAlternatively, remove some RPC methods (use more parameters per RPC maybe).\n\nAfter a RPC-list refresh, make sure you change the game version where you use `PhotonNetwork.ConnectUsingSettings()`."
- string `SkipRPCListUpdateLabel` = "Skip RPC-list update"
- string `PUNNameReplaceTitle` = "Warning: RPC-list Compatibility"
- string `PUNNameReplaceLabel` = "PUN replaces RPC names with numbers by using the RPC-list. All clients must use the same list for that.\n\nClearing it most likely makes your client incompatible with previous versions! Change your game version or make sure the RPC-list matches other clients."
- string `RPCListCleared` = "Clear RPC-list"
- string `ServerSettingsCleanedWarning` = "Cleared the PhotonServerSettings.RpcList! This makes new builds incompatible with older ones. Better change game version in `PhotonNetwork.ConnectUsingSettings()`."
- string `RpcFoundMessage` = "Some code uses the obsolete RPC attribute. PUN now requires the `PunRPC` attribute to mark remote-callable methods.\n\nThe Editor can search and replace that code which will modify your source."
- string `RpcFoundDialogTitle` = "RPC Attribute Outdated"
- string `RpcReplaceButton` = "Replace. I got a backup."
- string `RpcSkipReplace` = "Not now."
- string `WizardMainWindowInfo` = "This window should help you find important settings for PUN, as well as documentation."

8.71.1 Member Data Documentation

- 8.71.1.1 string `PunWizardText.AlreadyRegisteredInfo` = "The email is registered so we can't fetch your AppId (without password).\n\nPlease login online to get your AppId and paste it above."
- 8.71.1.2 string `PunWizardText.AppliedToSettingsInfo` = "Your AppId is now applied to this project."
- 8.71.1.3 string `PunWizardText.CancelButton` = "Cancel"
- 8.71.1.4 string `PunWizardText.CloseWindowButton` = "Close"
- 8.71.1.5 string `PunWizardText.ComparisonPageButton` = "Cloud versus OnPremise"
- 8.71.1.6 string `PunWizardText.ConnectionInfo` = "Connecting to the account service..."
- 8.71.1.7 string `PunWizardText.ConnectionTitle` = "Connecting"
- 8.71.1.8 string `PunWizardText.ConverterLabel` = "Converter"
- 8.71.1.9 string `PunWizardText.DocumentationLabel` = "Documentation"
- 8.71.1.10 string `PunWizardText.EmailOrAppIdLabel` = "AppId or Email"

- 8.71.1.11 string PunWizardText.ErrorTextTitle = "Error"
- 8.71.1.12 string PunWizardText.FullRPCListLabel = "Your project's RPC-list is too long for PUN.\n\nYou can change PUN's source to use short-typed RPC index. Look for comments 'LIMITS RPC COUNT'\n\nAlternatively, remove some RPC methods (use more parameters per RPC maybe).\n\nAfter a RPC-list refresh, make sure you change the game version where you use PhotonNetwork.ConnectUsingSettings()."
- 8.71.1.13 string PunWizardText.FullRPCListTitle = "Warning: RPC-list is full!"
- 8.71.1.14 string PunWizardText.IncorrectRPCListLabel = "Your project's RPC-list is full, so we can't add some RPCs just compiled.\n\nBy removing outdated RPCs, the list will be long enough but incompatible with older client builds!\n\nMake sure you change the game version where you use PhotonNetwork.ConnectUsingSettings()."
- 8.71.1.15 string PunWizardText.IncorrectRPCListTitle = "Warning: RPC-list becoming incompatible!"
- 8.71.1.16 string PunWizardText.LocateSettingsButton = "Locate PhotonServerSettings"
- 8.71.1.17 string PunWizardText.MainMenuButton = "Main Menu"
- 8.71.1.18 string PunWizardText.MobileExportNoteLabel = "Build for mobiles impossible. Get PUN+ or Unity Pro for mobile or use Unity 5."
- 8.71.1.19 string PunWizardText.MobilePunPlusExportNoteLabel = "PUN+ available. Using native sockets for iOS/Android."
- 8.71.1.20 string PunWizardText.OkButton = "Ok"
- 8.71.1.21 string PunWizardText.OpenCloudDashboardText = "Cloud Dashboard Login"
- 8.71.1.22 string PunWizardText.OpenCloudDashboardTooltip = "Review Cloud App information and statistics."
- 8.71.1.23 string PunWizardText.OpenDevNetText = "DevNet / Manual"
- 8.71.1.24 string PunWizardText.OpenDevNetTooltip = "Online documentation for Photon."
- 8.71.1.25 string PunWizardText.OpenForumText = "Open Forum"
- 8.71.1.26 string PunWizardText.OpenForumTooltip = "Online support for Photon."
- 8.71.1.27 string PunWizardText.OpenPDFText = "Reference PDF"
- 8.71.1.28 string PunWizardText.OpenPDFTooltip = "Opens the local documentation pdf."
- 8.71.1.29 string PunWizardText.OwnHostCloudCompareLabel = "I am not quite sure how 'my own host' compares to 'cloud'."
- 8.71.1.30 string PunWizardText.PUNNameReplaceLabel = "PUN replaces RPC names with numbers by using the RPC-list. All clients must use the same list for that.\n\nClearing it most likely makes your client incompatible with previous versions! Change your game version or make sure the RPC-list matches other clients."

- 8.71.1.31 string PunWizardText.PUNNameReplaceTitle = "Warning: RPC-list Compatibility"
- 8.71.1.32 string PunWizardText.PUNWizardLabel = "PUN Wizard"
- 8.71.1.33 string PunWizardText.RegisteredNewAccountInfo = "We created a (free) account and fetched you an Appld.\nWelcome. Your PUN project is setup."
- 8.71.1.34 string PunWizardText.RemoveOutdatedRPCsLabel = "Remove outdated RPCs"
- 8.71.1.35 string PunWizardText.RpcFoundDialogTitle = "RPC Attribute Outdated"
- 8.71.1.36 string PunWizardText.RpcFoundMessage = "Some code uses the obsolete RPC attribute. PUN now requires the **PunRPC** attribute to mark remote-callable methods.\nThe Editor can search and replace that code which will modify your source."
- 8.71.1.37 string PunWizardText.RPCListCleared = "Clear RPC-list"
- 8.71.1.38 string PunWizardText.RpcReplaceButton = "Replace. I got a backup."
- 8.71.1.39 string PunWizardText.RpcSkipReplace = "Not now."
- 8.71.1.40 string PunWizardText.ServerSettingsCleanedWarning = "Cleared the PhotonServerSettings.RpcList! This makes new builds incompatible with older ones. Better change game version in **PhotonNetwork.ConnectUsingSettings()**."
- 8.71.1.41 string PunWizardText.SettingsButton = "Settings"
- 8.71.1.42 string PunWizardText.SettingsHighlightLabel = "Highlights the used photon settings file in the project."
- 8.71.1.43 string PunWizardText.SetupButton = "Setup Project"
- 8.71.1.44 string PunWizardText.SetupCompleteInfo = "Done!\nAll connection settings can be edited in the PhotonServerSettings now.\nHave a look."
- 8.71.1.45 string PunWizardText.SetupServerCloudLabel = "Setup wizard for setting up your own server or the cloud."
- 8.71.1.46 string PunWizardText.SetupWizardInfo = "Thanks for importing Photon Unity Networking.\nThis window should set you up.\n\n- To use an existing Photon Cloud App, enter your Appld.\n- To register an account or access an existing one, enter the account's mail address.\n- To use Photon OnPremise, skip this step."
- 8.71.1.47 string PunWizardText.SetupWizardTitle = "PUN Setup"
- 8.71.1.48 string PunWizardText.SetupWizardWarningMessage = "You have not yet run the Photon setup wizard! Your game won't be able to connect. See Windows -> Photon Unity Networking."
- 8.71.1.49 string PunWizardText.SetupWizardWarningTitle = "Warning"
- 8.71.1.50 string PunWizardText.SkipButton = "Skip"

- 8.71.1.51 `string PunWizardText.SkipRegistrationInfo` = "Skipping? No problem:\nEdit your server settings in the PhotonServerSettings file."
- 8.71.1.52 `string PunWizardText.SkipRPCListUpdateLabel` = "Skip RPC-list update"
- 8.71.1.53 `string PunWizardText.StartButton` = "Start"
- 8.71.1.54 `string PunWizardText.UNtoPUNLabel` = "Converts pure Unity Networking to Photon Unity Networking."
- 8.71.1.55 `string PunWizardText.WarningPhotonDisconnect` = ""
- 8.71.1.56 `string PunWizardText.WindowTitle` = "PUN Wizard"
- 8.71.1.57 `string PunWizardText.WizardMainWindowInfo` = "This window should help you find important settings for PUN, as well as documentation."

The documentation for this class was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Editor/Photon↔ Network/[PhotonEditor.cs](#)

8.72 QuitOnEscapeOrBack Class Reference

Inherits MonoBehaviour.

The documentation for this class was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/UtilityScripts/[Quit↔ OnEscapeOrBack.cs](#)

8.73 RaiseEventOptions Class Reference

Aggregates several less-often used options for operation RaiseEvent. See field descriptions for usage details.

Public Attributes

- [EventCaching](#) [CachingOption](#)
Defines if the server should simply send the event, put it in the cache or remove events that are like this one.
- `byte` [InterestGroup](#)
The number of the Interest Group to send this to. 0 goes to all users but to get 1 and up, clients must subscribe to the group first.
- `int[]` [TargetActors](#)
A list of PhotonPlayer.IDs to send this event to. You can implement events that just go to specific users this way.
- [ReceiverGroup](#) [Receivers](#)
Sends the event to All, MasterClient or Others (default). Be careful with MasterClient, as the client might disconnect before it got the event and it gets lost.
- `byte` [SequenceChannel](#)
Events are ordered per "channel". If you have events that are independent of others, they can go into another sequence or channel.
- `bool` [ForwardToWebhook](#)
Events can be forwarded to Webhooks, which can evaluate and use the events to follow the game's state.
- `bool` [Encrypt](#)

Static Public Attributes

- static readonly [RaiseEventOptions Default](#) = new [RaiseEventOptions](#)()

Default options: CachingOption: DoNotCache, InterestGroup: 0, targetActors: null, receivers: Others, sequence↵Channel: 0.

8.73.1 Detailed Description

Aggregates several less-often used options for operation RaiseEvent. See field descriptions for usage details.

8.73.2 Member Data Documentation

8.73.2.1 EventCaching RaiseEventOptions.CachingOption

Defines if the server should simply send the event, put it in the cache or remove events that are like this one.

When using option: SliceSetIndex, SlicePurgeIndex or SlicePurgeUpToIndex, set a CacheSliceIndex. All other options except SequenceChannel get ignored.

8.73.2.2 readonly RaiseEventOptions RaiseEventOptions.Default = new RaiseEventOptions() [static]

Default options: CachingOption: DoNotCache, InterestGroup: 0, targetActors: null, receivers: Others, sequence↵Channel: 0.

8.73.2.3 bool RaiseEventOptions.Encrypt

8.73.2.4 bool RaiseEventOptions.ForwardToWebhook

Events can be forwarded to Webhooks, which can evaluate and use the events to follow the game's state.

8.73.2.5 byte RaiseEventOptions.InterestGroup

The number of the Interest Group to send this to. 0 goes to all users but to get 1 and up, clients must subscribe to the group first.

8.73.2.6 ReceiverGroup RaiseEventOptions.Receivers

Sends the event to All, MasterClient or Others (default). Be careful with MasterClient, as the client might disconnect before it got the event and it gets lost.

8.73.2.7 byte RaiseEventOptions.SequenceChannel

Events are ordered per "channel". If you have events that are independent of others, they can go into another sequence or channel.

8.73.2.8 `int [] RaiseEventOptions.TargetActors`

A list of PhotonPlayer.IDs to send this event to. You can implement events that just go to specific users this way.

The documentation for this class was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/Photon↵
Network/[LoadbalancingPeer.cs](#)

8.74 Region Class Reference

Public Member Functions

- override string [ToString](#) ()

Static Public Member Functions

- static [CloudRegionCode Parse](#) (string codeAsString)

Public Attributes

- [CloudRegionCode Code](#)
- string [HostAndPort](#)
- int [Ping](#)

8.74.1 Member Function Documentation

8.74.1.1 `static CloudRegionCode Region.Parse (string codeAsString)` [`static`]

8.74.1.2 `override string Region.ToString ()`

8.74.2 Member Data Documentation

8.74.2.1 `CloudRegionCode Region.Code`

8.74.2.2 `string Region.HostAndPort`

8.74.2.3 `int Region.Ping`

The documentation for this class was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/Photon↵
Network/[ServerSettings.cs](#)

8.75 Room Class Reference

This class resembles a room that PUN joins (or joined).

Inherits [RoomInfo](#).

Public Member Functions

- void [SetCustomProperties](#) ([Hashtable](#) propertiesToSet, [Hashtable](#) expectedValues=null, bool web↔Forward=false)
Updates the current room's Custom Properties with new/updated key-values.
- void [SetPropertiesListedInLobby](#) (string[] propsListedInLobby)
Enables you to define the properties available in the lobby if not all properties are needed to pick a room.
- override string [ToString](#) ()
Returns a summary of this [Room](#) instance as string.
- new string [ToStringFull](#) ()
Returns a summary of this [Room](#) instance as longer string, including Custom Properties.

Properties

- new int [playerCount](#) [get]
Count of players in this room.
- new string [name](#) [get, set]
The name of a room. Unique identifier (per Loadbalancing group) for a room/match.
- new int [maxPlayers](#) [get, set]
Sets a limit of players to this room.
- new bool [open](#) [get, set]
Defines if the room can be joined.
- new bool [visible](#) [get, set]
Defines if the room is listed in its lobby.
- string[] [propertiesListedInLobby](#) [get]
A list of custom properties that should be forwarded to the lobby and listed there.
- bool [autoCleanUp](#) [get]
Gets if this room uses autoCleanUp to remove all (buffered) RPCs and instantiated GameObjects when a player leaves.

Additional Inherited Members

8.75.1 Detailed Description

This class resembles a room that PUN joins (or joined).

The properties are settable as opposed to those of a [RoomInfo](#) and you can close or hide "your" room.

8.75.2 Member Function Documentation

8.75.2.1 `void Room.SetCustomProperties (Hashtable propertiesToSet, Hashtable expectedValues = null, bool webForward = false)`

Updates the current room's Custom Properties with new/updated key-values.

Custom Properties are a key-value set (Hashtable) which is available to all players in a room. They can relate to the room or individual players and are useful when only the current value of something is of interest. For example: The map of a room. All keys must be strings.

The [Room](#) and the [PhotonPlayer](#) class both have SetCustomProperties methods. Also, both classes offer access to current key-values by: customProperties.

Always use SetCustomProperties to change values. To reduce network traffic, set only values that actually changed. New properties are added, existing values are updated. Other values will not be changed, so only provide values that changed or are new.

To delete a named (custom) property of this room, use null as value.

Locally, SetCustomProperties will update it's cache without delay. Other clients are updated through [Photon](#) (the server) with a fitting operation.

Check and Swap

SetCustomProperties have the option to do a server-side Check-And-Swap (CAS): Values only get updated if the expected values are correct. The expectedValues can be different key/values than the propertiesToSet. So you can check some key and set another key's value (if the check succeeds).

If the client's knowledge of properties is wrong or outdated, it can't set values with CAS. This can be useful to keep players from concurrently setting values. For example: If all players try to pickup some card or item, only one should get it. With CAS, only the first SetProperty gets executed server-side and any other (sent at the same time) fails.

The server will broadcast successfully changed values and the local "cache" of customProperties only gets updated after a roundtrip (if anything changed).

You can do a "webForward": [Photon](#) will send the changed properties to a WebHook defined for your application.

OfflineMode

While [PhotonNetwork.offlineMode](#) is true, the expectedValues and webForward parameters are ignored. In Offline↔ Mode, the local customProperties values are immediately updated (without the roundtrip).

Parameters

<i>propertiesToSet</i>	The new properties to be set.
<i>expectedValues</i>	At least one property key/value set to check server-side. Key and value must be correct. Ignored in OfflineMode.
<i>webForward</i>	Set to true, to forward the set properties to a WebHook, defined for this app (in Dashboard). Ignored in OfflineMode.

8.75.2.2 `void Room.SetPropertiesListedInLobby (string[] propsListedInLobby)`

Enables you to define the properties available in the lobby if not all properties are needed to pick a room.

It makes sense to limit the amount of properties sent to users in the lobby as this improves speed and stability.

Parameters

<i>propsListedInLobby</i>	An array of custom room property names to forward to the lobby.
---------------------------	---

8.75.2.3 override string Room.ToString ()

Returns a summary of this [Room](#) instance as string.

Returns

Summary of this [Room](#) instance.

8.75.2.4 new string Room.ToStringFull ()

Returns a summary of this [Room](#) instance as longer string, including Custom Properties.

Returns

Summary of this [Room](#) instance.

8.75.3 Property Documentation

8.75.3.1 bool Room.autoCleanUp [get]

Gets if this room uses autoCleanUp to remove all (buffered) RPCs and instantiated GameObjects when a player leaves.

8.75.3.2 new int Room.maxPlayers [get], [set]

Sets a limit of players to this room.

This property is shown in lobby, too. If the room is full (players count == maxplayers), joining this room will fail.

8.75.3.3 new string Room.name [get], [set]

The name of a room. Unique identifier (per Loadbalancing group) for a room/match.

8.75.3.4 new bool Room.open [get], [set]

Defines if the room can be joined.

This does not affect listing in a lobby but joining the room will fail if not open. If not open, the room is excluded from random matchmaking. Due to racing conditions, found matches might become closed before they are joined. Simply re-connect to master and find another. Use property "visible" to not list the room.

8.75.3.5 `new int Room.playerCount` `[get]`

Count of players in this room.

8.75.3.6 `string [] Room.propertiesListedInLobby` `[get]`

A list of custom properties that should be forwarded to the lobby and listed there.

8.75.3.7 `new bool Room.visible` `[get], [set]`

Defines if the room is listed in its lobby.

Rooms can be created invisible, or changed to invisible. To change if a room can be joined, use property: open.

The documentation for this class was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/Photon↔ Network/[Room.cs](#)

8.76 RoomInfo Class Reference

A simplified room with just the info required to list and join, used for the room listing in the lobby.

Inherited by [Room](#).

Public Member Functions

- override bool [Equals](#) (object p)
Makes [RoomInfo](#) comparable (by name).
- override int [GetHashCode](#) ()
Accompanies Equals, using the name's GetHashCode as return.
- override string [ToString](#) ()
Simple printing in method.
- string [ToStringFull](#) ()
Simple printing in method.

Protected Attributes

- byte [maxPlayersField](#) = 0
Backing field for property.
- bool [openField](#) = true
Backing field for property.
- bool [visibleField](#) = true
Backing field for property.
- bool [autoCleanUpField](#) = [PhotonNetwork.autoCleanUpPlayerObjects](#)
Backing field for property. False unless the GameProperty is set to true (else it's not sent).
- string [nameField](#)
Backing field for property.

Properties

- bool [removedFromList](#) [get, set]
Used internally in lobby, to mark rooms that are no longer listed.
- [Hashtable](#) [customProperties](#) [get]
Read-only "cache" of custom properties of a room.
- string [name](#) [get]
The name of a room. Unique identifier (per Loadbalancing group) for a room/match.
- int [playerCount](#) [get]
Only used internally in lobby, to display number of players in room (while you're not in).
- bool [isLocalClientInside](#) [get, set]
State if the local client is already in the game or still going to join it on gameserver (in lobby always false).
- byte [maxPlayers](#) [get]
Sets a limit of players to this room.
- bool [open](#) [get]
Defines if the room can be joined.
- bool [visible](#) [get]
Defines if the room is listed in its lobby.

8.76.1 Detailed Description

A simplified room with just the info required to list and join, used for the room listing in the lobby.

The properties are not settable (open, maxPlayers, etc).

This class resembles info about available rooms, as sent by the Master server's lobby. Consider all values as readonly. None are synced (only updated by events by server).

8.76.2 Member Function Documentation

8.76.2.1 override bool RoomInfo.Equals (object p)

Makes [RoomInfo](#) comparable (by name).

8.76.2.2 override int RoomInfo.GetHashCode ()

Accompanies Equals, using the name's GetHashCode as return.

Returns

8.76.2.3 override string RoomInfo.ToString ()

Simple printingin method.

Returns

Summary of this [RoomInfo](#) instance.

8.76.2.4 string RoomInfo.ToStringFull ()

Simple printingin method.

Returns

Summary of this [RoomInfo](#) instance.

8.76.3 Member Data Documentation

8.76.3.1 bool RoomInfo.autoCleanUpField = PhotonNetwork.autoCleanUpPlayerObjects [protected]

Backing field for property. False unless the GameProperty is set to true (else it's not sent).

8.76.3.2 byte RoomInfo.maxPlayersField = 0 [protected]

Backing field for property.

8.76.3.3 string RoomInfo.nameField [protected]

Backing field for property.

8.76.3.4 bool RoomInfo.openField = true [protected]

Backing field for property.

8.76.3.5 bool RoomInfo.visibleField = true [protected]

Backing field for property.

8.76.4 Property Documentation

8.76.4.1 Hashtable RoomInfo.customProperties [get]

Read-only "cache" of custom properties of a room.

Set via [Room.SetCustomProperties](#) (not available for [RoomInfo](#) class!).

All keys are string-typed and the values depend on the game/application.

[Room.SetCustomProperties](#)

8.76.4.2 `bool RoomInfo.isLocalClientInside` `[get]`, `[set]`

State if the local client is already in the game or still going to join it on gameserver (in lobby always false).

8.76.4.3 `byte RoomInfo.maxPlayers` `[get]`

Sets a limit of players to this room.

This property is shown in lobby, too. If the room is full (players count == maxplayers), joining this room will fail.

As part of [RoomInfo](#) this can't be set. As part of a [Room](#) (which the player joined), the setter will update the server and all clients.

8.76.4.4 `string RoomInfo.name` `[get]`

The name of a room. Unique identifier (per Loadbalancing group) for a room/match.

8.76.4.5 `bool RoomInfo.open` `[get]`

Defines if the room can be joined.

This does not affect listing in a lobby but joining the room will fail if not open. If not open, the room is excluded from random matchmaking. Due to racing conditions, found matches might become closed before they are joined. Simply re-connect to master and find another. Use property "IsVisible" to not list the room.

As part of [RoomInfo](#) this can't be set. As part of a [Room](#) (which the player joined), the setter will update the server and all clients.

8.76.4.6 `int RoomInfo.playerCount` `[get]`

Only used internally in lobby, to display number of players in room (while you're not in).

8.76.4.7 `bool RoomInfo.removedFromList` `[get]`, `[set]`

Used internally in lobby, to mark rooms that are no longer listed.

8.76.4.8 `bool RoomInfo.visible` `[get]`

Defines if the room is listed in its lobby.

Rooms can be created invisible, or changed to invisible. To change if a room can be joined, use property: open.

As part of [RoomInfo](#) this can't be set. As part of a [Room](#) (which the player joined), the setter will update the server and all clients.

The documentation for this class was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/Photon↔ Network/[RoomInfo.cs](#)

8.77 RoomOptions Class Reference

Wraps up common room properties needed when you create rooms.

Public Attributes

- byte `maxPlayers`
Max number of players that can be in the room at any time. 0 means "no limit".
- `Hashtable` `customRoomProperties`
The room's custom properties to set.
- `string[]` `customRoomPropertiesForLobby` = `new string[0]`
Defines the custom room properties that get listed in the lobby.

Properties

- bool `isVisible` [get, set]
Defines if this room is listed in the lobby.
- bool `isOpen` [get, set]
Defines if this room can be joined at all.
- bool `cleanupCacheOnLeave` [get, set]
Time To Live (TTL) for an 'actor' in a room. If a client disconnects, this actor is inactive first and removed after this timeout. In milliseconds.
- bool `suppressRoomEvents` [get]
Tells the server to skip room events for joining and leaving players.

8.77.1 Detailed Description

Wraps up common room properties needed when you create rooms.

This directly maps to what the fields in the `Room` class.

8.77.2 Member Data Documentation

8.77.2.1 `Hashtable RoomOptions.customRoomProperties`

The room's custom properties to set.

Use string keys!

Custom room properties are any key-values you need to define the game's setup. The shorter your keys are, the better. Example: Map, Mode (could be "m" when used with "Map"), TileSet (could be "t").

8.77.2.2 `string [] RoomOptions.customRoomPropertiesForLobby = new string[0]`

Defines the custom room properties that get listed in the lobby.

Name the custom room properties that should be available to clients that are in a lobby. Use with care. Unless a custom property is essential for matchmaking or user info, it should not be sent to the lobby, which causes traffic and delays for clients in the lobby.

Default: No custom properties are sent to the lobby.

8.77.2.3 byte RoomOptions.maxPlayers

Max number of players that can be in the room at any time. 0 means "no limit".

8.77.3 Property Documentation

8.77.3.1 bool RoomOptions.cleanupCacheOnLeave [get], [set]

Time To Live (TTL) for an 'actor' in a room. If a client disconnects, this actor is inactive first and removed after this timeout. In milliseconds.

Time To Live (TTL) for a room when the last player leaves. Keeps room in memory for case a player re-joins soon. In milliseconds.

Time To Live (TTL) for a room when the last player leaves. Keeps room in memory for case a player re-joins soon. In milliseconds.

Removes a user's events and properties from the room when a user leaves.

This makes sense when in rooms where players can't place items in the room and just vanish entirely. When you disable this, the event history can become too long to load if the room stays in use indefinitely. Default: true. Cleans up the cache and props of leaving users.

8.77.3.2 bool RoomOptions.isOpen [get], [set]

Defines if this room can be joined at all.

If a room is closed, no player can join this. As example this makes sense when 3 of 4 possible players start their gameplay early and don't want anyone to join during the game. The room can still be listed in the lobby (set isVisible to control lobby-visibility).

8.77.3.3 bool RoomOptions.isVisible [get], [set]

Defines if this room is listed in the lobby.

If not, it also is not joined randomly.

A room that is not visible will be excluded from the room lists that are sent to the clients in lobbies. An invisible room can be joined by name but is excluded from random matchmaking.

Use this to "hide" a room and simulate "private rooms". Players can exchange a roomname and create it invisible to avoid anyone else joining it.

8.77.3.4 bool RoomOptions.suppressRoomEvents [get]

Tells the server to skip room events for joining and leaving players.

Using this makes the client unaware of the other players in a room. That can save some traffic if you have some server logic that updates players but it can also limit the client's usability.

PUN will break if you use this, so it's not settable.

The documentation for this class was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/Photon↔ Network/[PhotonClasses.cs](#)

8.78 UnityEngine.SceneManagement.SceneManager Class Reference

Minimal implementation of the [SceneManager](#) for older Unity, up to v5.2.

Static Public Member Functions

- static void [LoadScene](#) (string name)
- static void [LoadScene](#) (int buildIndex)

8.78.1 Detailed Description

Minimal implementation of the [SceneManager](#) for older Unity, up to v5.2.

8.78.2 Member Function Documentation

8.78.2.1 static void UnityEngine.SceneManagement.SceneManager.LoadScene (string *name*) [static]

8.78.2.2 static void UnityEngine.SceneManagement.SceneManager.LoadScene (int *buildIndex*) [static]

The documentation for this class was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/Photon↔ Network/[PhotonClasses.cs](#)

8.79 SceneManagerHelper Class Reference

Properties

- static string [ActiveSceneName](#) [get]
- static int [ActiveSceneBuildIndex](#) [get]

8.79.1 Property Documentation

8.79.1.1 int SceneManagerHelper.ActiveSceneBuildIndex [static], [get]

8.79.1.2 string SceneManagerHelper.ActiveSceneName [static], [get]

The documentation for this class was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/Photon↔ Network/[PhotonClasses.cs](#)

8.80 SceneSetting Class Reference

Public Attributes

- string [sceneName](#)
- int [minViewId](#)

8.80.1 Member Data Documentation

8.80.1.1 int SceneSetting.minViewId

8.80.1.2 string SceneSetting.sceneName

The documentation for this class was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Editor/PhotonNetwork/[PunSceneSettings.cs](#)

8.81 ScoreExtensions Class Reference

Static Public Member Functions

- static void [SetScore](#) (this [PhotonPlayer](#) player, int newScore)
- static void [AddScore](#) (this [PhotonPlayer](#) player, int scoreToAddToCurrent)
- static int [GetScore](#) (this [PhotonPlayer](#) player)

8.81.1 Member Function Documentation

8.81.1.1 static void ScoreExtensions.AddScore (this [PhotonPlayer](#) *player*, int *scoreToAddToCurrent*) [static]

8.81.1.2 static int ScoreExtensions.GetScore (this [PhotonPlayer](#) *player*) [static]

8.81.1.3 static void ScoreExtensions.SetScore (this [PhotonPlayer](#) *player*, int *newScore*) [static]

The documentation for this class was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/UtilityScripts/[PhotonPlayerScores.cs](#)

8.82 ServerSettings Class Reference

Collection of connection-relevant settings, used internally by [PhotonNetwork.ConnectUsingSettings](#).

Inherits [ScriptableObject](#).

Public Types

- enum [HostingOption](#) {
[HostingOption.NotSet](#) = 0, [HostingOption.PhotonCloud](#) = 1, [HostingOption.SelfHosted](#) = 2, [HostingOption.OfflineMode](#) = 3,
[HostingOption.BestRegion](#) = 4 }

Public Member Functions

- void [UseCloudBestRegion](#) (string cloudAppid)
- void [UseCloud](#) (string cloudAppid)
- void [UseCloud](#) (string cloudAppid, [CloudRegionCode](#) code)
- void [UseMyServer](#) (string serverAddress, int serverPort, string application)
- override string [ToString](#) ()

Public Attributes

- [HostingOption](#) [HostType](#) = [HostingOption.NotSet](#)
- ConnectionProtocol [Protocol](#) = ConnectionProtocol.Udp
- string [ServerAddress](#) = ""
- int [ServerPort](#) = 5055
- string [AppID](#) = ""
- [CloudRegionCode](#) [PreferredRegion](#)
- [CloudRegionFlag](#) [EnabledRegions](#) = ([CloudRegionFlag](#))(-1)
- bool [JoinLobby](#)
- bool [EnableLobbyStatistics](#)
- List< string > [RpcList](#) = new List<string>()
- bool [DisableAutoOpenWizard](#)

8.82.1 Detailed Description

Collection of connection-relevant settings, used internally by [PhotonNetwork.ConnectUsingSettings](#).

8.82.2 Member Enumeration Documentation

8.82.2.1 enum [ServerSettings.HostingOption](#) [strong]

Enumerator

NotSet

PhotonCloud

SelfHosted

OfflineMode

BestRegion

8.82.3 Member Function Documentation

8.82.3.1 `override string ServerSettings.ToString ()`

8.82.3.2 `void ServerSettings.UseCloud (string cloudAppid)`

8.82.3.3 `void ServerSettings.UseCloud (string cloudAppid, CloudRegionCode code)`

8.82.3.4 `void ServerSettings.UseCloudBestRegion (string cloudAppid)`

8.82.3.5 `void ServerSettings.UseMyServer (string serverAddress, int serverPort, string application)`

8.82.4 Member Data Documentation

8.82.4.1 `string ServerSettings.AppID = ""`

8.82.4.2 `bool ServerSettings.DisableAutoOpenWizard`

8.82.4.3 `CloudRegionFlag ServerSettings.EnabledRegions = (CloudRegionFlag)(-1)`

8.82.4.4 `bool ServerSettings.EnableLobbyStatistics`

8.82.4.5 `HostingOption ServerSettings.HostType = HostingOption.NotSet`

8.82.4.6 `bool ServerSettings.JoinLobby`

8.82.4.7 `CloudRegionCode ServerSettings.PreferredRegion`

8.82.4.8 `ConnectionProtocol ServerSettings.Protocol = ConnectionProtocol.Udp`

8.82.4.9 `List<string> ServerSettings.RpcList = new List<string>()`

8.82.4.10 `string ServerSettings.ServerAddress = ""`

8.82.4.11 `int ServerSettings.ServerPort = 5055`

The documentation for this class was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/Photon↔ Network/[ServerSettings.cs](#)

8.83 ServerSettingsInspector Class Reference

Inherits Editor.

Public Types

- enum [ProtocolChoices](#) { [ProtocolChoices.Udp](#) = ConnectionProtocol.Udp, [ProtocolChoices.Tcp](#) = ConnectionProtocol.Tcp }

Public Member Functions

- override void [OnInspectorGUI](#) ()

Static Public Member Functions

- static bool [IsAppld](#) (string val)
Checks if a string is a Guid by attempting to create one.

8.83.1 Member Enumeration Documentation

8.83.1.1 enum [ServerSettingsInspector.ProtocolChoices](#) [strong]

Enumerator

Udp

Tcp

8.83.2 Member Function Documentation

8.83.2.1 static bool [ServerSettingsInspector.IsAppld](#) (string val) [static]

Checks if a string is a Guid by attempting to create one.

Parameters

<i>val</i>	The potential guid to check.
------------	------------------------------

Returns

True if new Guid(val) did not fail.

8.83.2.2 override void [ServerSettingsInspector.OnInspectorGUI](#) ()

The documentation for this class was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Editor/Photon↔ Network/[ServerSettingsInspector.cs](#)

8.84 ServerTime Class Reference

Inherits MonoBehaviour.

The documentation for this class was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/UtilityScripts/[ServerTime.cs](#)

8.85 ShowInfoOfPlayer Class Reference

Can be attached to a GameObject to show info about the owner of the [PhotonView](#).

Inherits [Photon.MonoBehaviour](#).

Public Attributes

- float [CharacterSize](#) = 0
- Font [font](#)
- bool [DisableOnOwnObjects](#)

Additional Inherited Members

8.85.1 Detailed Description

Can be attached to a GameObject to show info about the owner of the [PhotonView](#).

This is a Photon.Monobehaviour, which adds the property photonView (that's all).

8.85.2 Member Data Documentation

8.85.2.1 float ShowInfoOfPlayer.CharacterSize = 0

8.85.2.2 bool ShowInfoOfPlayer.DisableOnOwnObjects

8.85.2.3 Font ShowInfoOfPlayer.font

The documentation for this class was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/UtilityScripts/[ShowInfoOfPlayer.cs](#)

8.86 ShowStatusWhenConnecting Class Reference

Inherits MonoBehaviour.

Public Attributes

- GUISkin [Skin](#)

8.86.1 Member Data Documentation

8.86.1.1 GUISkin ShowStatusWhenConnecting.Skin

The documentation for this class was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/UtilityScripts/[Show↔StatusWhenConnecting.cs](#)

8.87 SmoothSyncMovement Class Reference

Inherits [Photon.MonoBehaviour](#).

Public Member Functions

- void [Awake](#) ()
- void [OnPhotonSerializeView](#) ([PhotonStream](#) stream, [PhotonMessageInfo](#) info)
- void [Update](#) ()

Public Attributes

- float [SmoothingDelay](#) = 5

Additional Inherited Members

8.87.1 Member Function Documentation

8.87.1.1 void SmoothSyncMovement.Awake ()

8.87.1.2 void SmoothSyncMovement.OnPhotonSerializeView ([PhotonStream](#) stream, [PhotonMessageInfo](#) info)

8.87.1.3 void SmoothSyncMovement.Update ()

8.87.2 Member Data Documentation

8.87.2.1 float SmoothSyncMovement.SmoothingDelay = 5

The documentation for this class was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/UtilityScripts/[Smooth↔SyncMovement.cs](#)

8.88 SupportLogger Class Reference

Inherits MonoBehaviour.

Public Member Functions

- void [Start](#) ()

Public Attributes

- bool [LogTrafficStats](#) = true

8.88.1 Member Function Documentation

8.88.1.1 void SupportLogger.Start ()

8.88.2 Member Data Documentation

8.88.2.1 bool SupportLogger.LogTrafficStats = true

The documentation for this class was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/UtilityScripts/[SupportLogger.cs](#)

8.89 SupportLogging Class Reference

Inherits MonoBehaviour.

Public Member Functions

- void [Start](#) ()
- void [OnApplicationQuit](#) ()
- void [LogStats](#) ()
- void [OnConnectedToPhoton](#) ()
- void [OnFailedToConnectToPhoton](#) ([DisconnectCause](#) cause)
- void [OnJoinedLobby](#) ()
- void [OnJoinedRoom](#) ()
- void [OnCreatedRoom](#) ()
- void [OnLeftRoom](#) ()
- void [OnDisconnectedFromPhoton](#) ()

Public Attributes

- bool [LogTrafficStats](#)

Protected Member Functions

- void [OnApplicationPause](#) (bool pause)

8.89.1 Member Function Documentation

8.89.1.1 void [SupportLogging.LogStats](#) ()

8.89.1.2 void [SupportLogging.OnApplicationPause](#) (bool *pause*) [protected]

8.89.1.3 void [SupportLogging.OnApplicationQuit](#) ()

8.89.1.4 void [SupportLogging.OnConnectedToPhoton](#) ()

8.89.1.5 void [SupportLogging.OnCreatedRoom](#) ()

8.89.1.6 void [SupportLogging.OnDisconnectedFromPhoton](#) ()

8.89.1.7 void [SupportLogging.OnFailedToConnectToPhoton](#) ([DisconnectCause](#) *cause*)

8.89.1.8 void [SupportLogging.OnJoinedLobby](#) ()

8.89.1.9 void [SupportLogging.OnJoinedRoom](#) ()

8.89.1.10 void [SupportLogging.OnLeftRoom](#) ()

8.89.1.11 void [SupportLogging.Start](#) ()

8.89.2 Member Data Documentation

8.89.2.1 bool [SupportLogging.LogTrafficStats](#)

The documentation for this class was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/UtilityScripts/[Support↔
Logger.cs](#)

8.90 PhotonAnimatorView.SynchronizedLayer Class Reference

Public Attributes

- [SynchronizeType](#) [SynchronizeType](#)
- int [LayerIndex](#)

8.90.1 Member Data Documentation

8.90.1.1 `int PhotonAnimatorView.SynchronizedLayer.LayerIndex`

8.90.1.2 `SynchronizeType PhotonAnimatorView.SynchronizedLayer.SynchronizeType`

The documentation for this class was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/Photon↔
Network/Views/[PhotonAnimatorView.cs](#)

8.91 PhotonAnimatorView.SynchronizedParameter Class Reference

Public Attributes

- [ParameterType](#) `Type`
- [SynchronizeType](#) `SynchronizeType`
- `string` [Name](#)

8.91.1 Member Data Documentation

8.91.1.1 `string PhotonAnimatorView.SynchronizedParameter.Name`

8.91.1.2 `SynchronizeType PhotonAnimatorView.SynchronizedParameter.SynchronizeType`

8.91.1.3 `ParameterType PhotonAnimatorView.SynchronizedParameter.Type`

The documentation for this class was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/Photon↔
Network/Views/[PhotonAnimatorView.cs](#)

8.92 TeamExtensions Class Reference

Extension used for [PunTeams](#) and [PhotonPlayer](#) class. Wraps access to the player's custom property.

Static Public Member Functions

- static [PunTeams.Team](#) [GetTeam](#) (this [PhotonPlayer](#) player)
Extension for [PhotonPlayer](#) class to wrap up access to the player's custom property.
- static void [SetTeam](#) (this [PhotonPlayer](#) player, [PunTeams.Team](#) team)
Switch that player's team to the one you assign.

8.92.1 Detailed Description

Extension used for [PunTeams](#) and [PhotonPlayer](#) class. Wraps access to the player's custom property.

8.92.2 Member Function Documentation

8.92.2.1 static [PunTeams.Team](#) TeamExtensions.GetTeam (this [PhotonPlayer](#) *player*) [static]

Extension for [PhotonPlayer](#) class to wrap up access to the player's custom property.

Returns

[PunTeam.Team](#).none if no team was found (yet).

8.92.2.2 static void TeamExtensions.SetTeam (this [PhotonPlayer](#) *player*, [PunTeams.Team](#) *team*) [static]

Switch that player's team to the one you assign.

Internally checks if this player is in that team already or not. Only team switches are actually sent.

Parameters

<i>player</i>	
<i>team</i>	

The documentation for this class was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/UtilityScripts/[PunTeams.cs](#)

8.93 ExitGames.Client.DemoParticle.TimeKeeper Class Reference

A utility class that turns it's ShouldExecute property to true after a set interval time has passed.

Public Member Functions

- [TimeKeeper](#) (int interval)
Creates a new [TimeKeeper](#) and sets it's interval.
- void [Reset](#) ()
ShouldExecute becomes false and the time interval is refreshed for next execution.

Properties

- int `Interval` [get, set]
Interval in which ShouldExecute should be true (and something is executed).
- bool `IsEnabled` [get, set]
A disabled `TimeKeeper` never turns ShouldExecute to true. Reset won't affect IsEnabled!
- bool `ShouldExecute` [get, set]
Turns true of the time interval has passed (after reset or creation) or someone set ShouldExecute manually.

8.93.1 Detailed Description

A utility class that turns it's ShouldExecute property to true after a set interval time has passed.

TimeKeepers can be useful to execute tasks in a certain interval within a game loop (integrating a recurring task into a certain thread).

An interval can be overridden, when you set ShouldExecute to true. Call Reset after execution of whatever you do to re-enable the `TimeKeeper` (ShouldExecute becomes false until interval passed). Being based on `Environment.TickCount`, this is not very precise but cheap.

8.93.2 Constructor & Destructor Documentation

8.93.2.1 `ExitGames.Client.DemoParticle.TimeKeeper.TimeKeeper (int interval)`

Creates a new `TimeKeeper` and sets it's interval.

Parameters

<code>interval</code>	
-----------------------	--

8.93.3 Member Function Documentation

8.93.3.1 `void ExitGames.Client.DemoParticle.TimeKeeper.Reset ()`

ShouldExecute becomes false and the time interval is refreshed for next execution.

Does not affect IsEnabled.

8.93.4 Property Documentation

8.93.4.1 `int ExitGames.Client.DemoParticle.TimeKeeper.Interval [get], [set]`

Interval in which ShouldExecute should be true (and something is executed).

8.93.4.2 `bool ExitGames.Client.DemoParticle.TimeKeeper.IsEnabled` `[get]`, `[set]`

A disabled [TimeKeeper](#) never turns ShouldExecute to true. Reset won't affect IsEnabled!

8.93.4.3 `bool ExitGames.Client.DemoParticle.TimeKeeper.ShouldExecute` `[get]`, `[set]`

Turns true if the time interval has passed (after reset or creation) or someone set ShouldExecute manually.

Call Reset to start a new interval.

The documentation for this class was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/UtilityScripts/[TimeKeeper.cs](#)

8.94 TypedLobby Class Reference

Refers to a specific lobby (and type) on the server.

Inherited by [TypedLobbyInfo](#).

Public Member Functions

- [TypedLobby](#) ()
- [TypedLobby](#) (string name, [LobbyType](#) type)
- override string [ToString](#) ()

Public Attributes

- string [Name](#)
Name of the lobby this game gets added to. Default: null, attached to default lobby. Lobbies are unique per lobbyName plus lobbyType, so the same name can be used when several types are existing.
- [LobbyType](#) [Type](#)
Type of the (named)lobby this game gets added to

Static Public Attributes

- static readonly [TypedLobby](#) [Default](#) = new [TypedLobby](#)()

Properties

- bool [IsDefault](#) `[get]`

8.94.1 Detailed Description

Refers to a specific lobby (and type) on the server.

The name and type are the unique identifier for a lobby.

Join a lobby via [PhotonNetwork.JoinLobby\(TypedLobby lobby\)](#).

The current lobby is stored in [PhotonNetwork.lobby](#).

8.94.2 Constructor & Destructor Documentation

8.94.2.1 `TypedLobby.TypedLobby ()`

8.94.2.2 `TypedLobby.TypedLobby (string name, LobbyType type)`

8.94.3 Member Function Documentation

8.94.3.1 `override string TypedLobby.ToString ()`

8.94.4 Member Data Documentation

8.94.4.1 `readonly TypedLobby TypedLobby.Default = new TypedLobby() [static]`

8.94.4.2 `string TypedLobby.Name`

Name of the lobby this game gets added to. Default: null, attached to default lobby. Lobbies are unique per lobbyName plus lobbyType, so the same name can be used when several types are existing.

8.94.4.3 `LobbyType TypedLobby.Type`

Type of the (named)lobby this game gets added to

8.94.5 Property Documentation

8.94.5.1 `bool TypedLobby.IsDefault [get]`

The documentation for this class was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/Photon↔ Network/[LoadbalancingPeer.cs](#)

8.95 TypedLobbyInfo Class Reference

Inherits [TypedLobby](#).

Public Member Functions

- override string [ToString](#) ()

Public Attributes

- int [PlayerCount](#)
- int [RoomCount](#)

Additional Inherited Members

8.95.1 Member Function Documentation

8.95.1.1 override string TypedLobbyInfo.ToString ()

8.95.2 Member Data Documentation

8.95.2.1 int TypedLobbyInfo.PlayerCount

8.95.2.2 int TypedLobbyInfo.RoomCount

The documentation for this class was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/Photon↵
Network/[LoadbalancingPeer.cs](#)

8.96 WebRpcResponse Class Reference

Reads an operation response of a WebRpc and provides convenient access to most common values.

Public Member Functions

- [WebRpcResponse](#) (OperationResponse response)
An OperationResponse for a WebRpc is needed to read it's values.
- string [ToStringFull](#) ()
Turns the response into an easier to read string.

Properties

- string [Name](#) [get]
Name of the WebRpc that was called.
- int [ReturnCode](#) [get]
ReturnCode of the WebService that answered the WebRpc.
- string [DebugMessage](#) [get]
Might be empty or null.
- Dictionary< string, object > [Parameters](#) [get]
Other key/values returned by the webservice that answered the WebRpc.

8.96.1 Detailed Description

Reads an operation response of a WebRpc and provides convenient access to most common values.

See method [PhotonNetwork.WebRpc](#).

Create a [WebRpcResponse](#) to access common result values.

The operationResponse.OperationCode should be: `OperationCode.WebRpc`.

8.96.2 Constructor & Destructor Documentation

8.96.2.1 `WebRpcResponse.WebRpcResponse (OperationResponse response)`

An OperationResponse for a WebRpc is needed to read it's values.

8.96.3 Member Function Documentation

8.96.3.1 `string WebRpcResponse.ToStringFull ()`

Turns the response into an easier to read string.

Returns

String resembling the result.

8.96.4 Property Documentation

8.96.4.1 `string WebRpcResponse.DebugMessage [get]`

Might be empty or null.

8.96.4.2 `string WebRpcResponse.Name [get]`

Name of the WebRpc that was called.

8.96.4.3 `Dictionary<string, object> WebRpcResponse.Parameters [get]`

Other key/values returned by the webservice that answered the WebRpc.

8.96.4.4 `int WebRpcResponse.ReturnCode [get]`

ReturnCode of the WebService that answered the WebRpc.

0 is commonly used to signal success.

-1 tells you: Got no ReturnCode from WebRpc service.

Other ReturnCodes are defined by the individual WebRpc and service.

The documentation for this class was generated from the following file:

- C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/Photon↔ Network/[PhotonClasses.cs](#)

Chapter 9

File Documentation

9.1 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/↔ Editor/PhotonNetwork/AccountService.cs File Reference

Classes

- class [AccountService](#)

9.2 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/↔ Editor/PhotonNetwork/PhotonConverter.cs File Reference

Classes

- class [PhotonConverter](#)

9.3 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/↔ Editor/PhotonNetwork/PhotonEditor.cs File Reference

Classes

- class [PunWizardText](#)
- class [PhotonEditor](#)

9.4 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/↔ Editor/PhotonNetwork/PhotonGUI.cs File Reference

Classes

- class [PhotonGUI](#)

9.5 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/↔ Editor/PhotonNetwork/PhotonViewHandler.cs File Reference

Classes

- class [PhotonViewHandler](#)

Typedefs

- using [Debug](#) = UnityEngine.Debug

9.5.1 Typedef Documentation

9.5.1.1 using Debug = UnityEngine.Debug

9.6 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/↔ Editor/PhotonNetwork/PhotonViewInspector.cs File Reference

Classes

- class [PhotonViewInspector](#)

9.7 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/↔ Editor/PhotonNetwork/PhotonViewPrefabApply.cs File Reference

9.8 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/↔ Editor/PhotonNetwork/PunSceneSettings.cs File Reference

Classes

- class [SceneSetting](#)
- class [PunSceneSettings](#)

9.9 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/↔ Editor/PhotonNetwork/ReorderableListResources.cs File Reference

Classes

- class **Rotorz.ReorderableList.Internal.ReorderableListResources**
Resources to assist with reorderable list control.

Namespaces

- namespace [Rotorz.ReorderableList.Internal](#)

9.10 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/↔ Editor/PhotonNetwork/ServerSettingsInspector.cs File Reference

Classes

- class [ServerSettingsInspector](#)

9.11 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/↔ Editor/PhotonNetwork/Views/PhotonAnimatorViewEditor.cs File Reference

Classes

- class [PhotonAnimatorViewEditor](#)

9.12 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/↔ Editor/PhotonNetwork/Views/PhotonRigidbody2DViewEditor.cs File Reference

Classes

- class [PhotonRigidbody2DViewEditor](#)

9.13 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/↔ Editor/PhotonNetwork/Views/PhotonRigidbodyViewEditor.cs File Reference

Classes

- class [PhotonRigidbodyViewEditor](#)

9.14 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/↔ Editor/PhotonNetwork/Views/PhotonTransformViewEditor.cs File Reference

Classes

- class [PhotonTransformViewEditor](#)

9.15 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/PhotonNetwork/CustomTypes.cs File Reference ↩

Sets up support for Unity-specific types.

Classes

- class **CustomTypes**

Internally used class, containing de/serialization methods for various Unity-specific classes.

9.15.1 Detailed Description

Sets up support for Unity-specific types.

Can be a blueprint how to register your own Custom Types for sending.

9.16 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/PhotonNetwork/Enums.cs File Reference ↩

Wraps up several of the commonly used enumerations.

Enumerations

- enum **PhotonNetworkingMessage** {
PhotonNetworkingMessage.OnConnectedToPhoton, PhotonNetworkingMessage.OnLeftRoom, PhotonNetworkingMessage.OnMasterClientSwitched, PhotonNetworkingMessage.OnPhotonCreateRoomFailed, PhotonNetworkingMessage.OnPhotonJoinRoomFailed, PhotonNetworkingMessage.OnCreatedRoom, PhotonNetworkingMessage.OnJoinedLobby, PhotonNetworkingMessage.OnLeftLobby, PhotonNetworkingMessage.OnDisconnectedFromPhoton, PhotonNetworkingMessage.OnConnectionFail, PhotonNetworkingMessage.OnFailedToConnectToPhoton, PhotonNetworkingMessage.OnReceivedRoomListUpdate, PhotonNetworkingMessage.OnJoinedRoom, PhotonNetworkingMessage.OnPhotonPlayerConnected, PhotonNetworkingMessage.OnPhotonPlayerDisconnected, PhotonNetworkingMessage.OnPhotonRandomJoinFailed, PhotonNetworkingMessage.OnConnectedToMaster, PhotonNetworkingMessage.OnPhotonSerializeView, PhotonNetworkingMessage.OnPhotonInstantiate, PhotonNetworkingMessage.OnPhotonMaxCccuReached, PhotonNetworkingMessage.OnPhotonCustomRoomPropertiesChanged, PhotonNetworkingMessage.OnPhotonPlayerPropertiesChanged, PhotonNetworkingMessage.OnUpdatedFriendList, PhotonNetworkingMessage.OnCustomAuthenticationFailed, PhotonNetworkingMessage.OnWebRpcResponse, PhotonNetworkingMessage.OnOwnershipRequest, PhotonNetworkingMessage.OnLobbyStatisticsUpdate }

This enum defines the set of MonoMessages Photon Unity Networking is using as callbacks.

- enum **PhotonLogLevel** { PhotonLogLevel.ErrorsOnly, PhotonLogLevel.Informational, PhotonLogLevel.Full }

Used to define the level of logging output created by the PUN classes.

- enum **PhotonTargets** {
PhotonTargets.All, PhotonTargets.Others, PhotonTargets.MasterClient, PhotonTargets.AllBuffered, PhotonTargets.OthersBuffered, PhotonTargets.AllViaServer, PhotonTargets.AllBufferedViaServer }

Enum of "target" options for RPCs.

- enum [CloudRegionCode](#) {
[CloudRegionCode.eu](#) = 0, [CloudRegionCode.us](#) = 1, [CloudRegionCode.asia](#) = 2, [CloudRegionCode.jp](#) = 3,
[CloudRegionCode.au](#) = 5, [CloudRegionCode.none](#) = 4 }
Currently available [Photon Cloud regions](#) as enum.
- enum [CloudRegionFlag](#) {
[CloudRegionFlag.eu](#) = 1 << 0, [CloudRegionFlag.us](#) = 1 << 1, [CloudRegionFlag.asia](#) = 1 << 2, [CloudRegionFlag.jp](#) = 1 << 3,
[CloudRegionFlag.au](#) = 1 << 4 }
Available regions as enum of flags.
- enum [ServerConnection](#) { [ServerConnection.MasterServer](#), [ServerConnection.GameServer](#), [ServerConnection.NameServer](#) }
Available server (types) for internally used field: server.
- enum [ConnectionState](#) {
[ConnectionState.Disconnected](#), [ConnectionState.Connecting](#), [ConnectionState.Connected](#), [ConnectionState.Disconnecting](#),
[ConnectionState.InitializingApplication](#) }
High level connection state of the client.
- enum [PeerState](#) {
[PeerState.Uninitialized](#), [PeerState.PeerCreated](#), [PeerState.Queued](#), [PeerState.Authenticated](#),
[PeerState.JoinedLobby](#), [PeerState.DisconnectingFromMasterserver](#), [PeerState.ConnectingToGameserver](#),
[PeerState.ConnectedToGameserver](#),
[PeerState.Joining](#), [PeerState.Joined](#), [PeerState.Leaving](#), [PeerState.DisconnectingFromGameserver](#),
[PeerState.ConnectingToMasterserver](#), [PeerState.QueuedComingFromGameserver](#), [PeerState.Disconnecting](#),
[PeerState.Disconnected](#),
[PeerState.ConnectedToMaster](#), [PeerState.ConnectingToNameServer](#), [PeerState.ConnectedToNameServer](#),
[PeerState.DisconnectingFromNameServer](#),
[PeerState.Authenticating](#) }
Detailed connection / networking peer state.
- enum [DisconnectCause](#) {
[DisconnectCause.ExceptionOnConnect](#) = [StatusCode.ExceptionOnConnect](#), [DisconnectCause.SecurityExceptionOnConnect](#) = [StatusCode.SecurityExceptionOnConnect](#), [DisconnectCause.TimeoutDisconnect](#) = [StatusCode.TimeoutDisconnect](#), [DisconnectCause.DisconnectByClientTimeout](#) = [StatusCode.TimeoutDisconnect](#),
[DisconnectCause.InternalReceiveException](#) = [StatusCode.ExceptionOnReceive](#), [DisconnectCause.DisconnectByServer](#) = [StatusCode.DisconnectByServer](#), [DisconnectCause.DisconnectByServerTimeout](#) = [StatusCode.DisconnectByServer](#), [DisconnectCause.DisconnectByServerLogic](#) = [StatusCode.DisconnectByServerLogic](#),
[DisconnectCause.DisconnectByServerUserLimit](#) = [StatusCode.DisconnectByServerUserLimit](#), [DisconnectCause.Exception](#) = [StatusCode.Exception](#), [DisconnectCause.InvalidRegion](#) = [ErrorCode.InvalidRegion](#),
[DisconnectCause.MaxCcuReached](#) = [ErrorCode.MaxCcuReached](#),
[DisconnectCause.InvalidAuthentication](#) = [ErrorCode.InvalidAuthentication](#), [DisconnectCause.AuthenticationTicketExpired](#) = 32753 }
Summarizes the cause for a disconnect.

9.16.1 Detailed Description

Wraps up several of the commonly used enumerations.

9.16.2 Enumeration Type Documentation

9.16.2.1 enum [CloudRegionCode](#) [strong]

Currently available [Photon Cloud regions](#) as enum.

This is used in [PhotonNetwork.ConnectToRegion](#).

Enumerator

eu European servers in Amsterdam.
us US servers (East Coast).
asia Asian servers in Singapore.
jp Japanese servers in Tokyo.
au Australian servers in Melbourne.
none No region selectedcs.

9.16.2.2 enum **CloudRegionFlag** [strong]

Available regions as enum of flags.

To be used as "enabled" flags for Best [Region](#) ping.

Note that these enum values skip [CloudRegionCode.none](#) and their values are in strict order (power of 2).

Enumerator

eu
us
asia
jp
au

9.16.2.3 enum **ConnectionState** [strong]

High level connection state of the client.

Better use the more detailed [PeerState](#).

Enumerator

Disconnected
Connecting
Connected
Disconnecting
InitializingApplication

9.16.2.4 enum **ServerConnection** [strong]

Available server (types) for internally used field: server.

[Photon](#) uses 3 different roles of servers: Name Server, Master Server and Game Server.

Enumerator

MasterServer This server is where matchmaking gets done and where clients can get lists of rooms in lobbies.
GameServer This server handles a number of rooms to execute and relay the messages between players (in a room).
NameServer This server is used initially to get the address (IP) of a Master Server for a specific region. Not used for [Photon](#) OnPremise (self hosted).

9.17 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/PhotonNetwork/Extensions.cs File Reference ↩

Classes

- class [Extensions](#)
This static class defines some useful extension methods for several existing classes (e.g.
- class [GameObjectExtensions](#)
Small number of extension methods that make it easier for PUN to work cross-Unity-versions.

Typedefs

- using [Hashtable](#) = ExitGames.Client.Photon.Hashtable
- using [SupportClass](#) = ExitGames.Client.Photon.SupportClass

9.17.1 Typedef Documentation

9.17.1.1 using [Hashtable](#) = ExitGames.Client.Photon.Hashtable

9.17.1.2 using [SupportClass](#) = ExitGames.Client.Photon.SupportClass

9.18 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/PhotonNetwork/FriendInfo.cs File Reference ↩

Classes

- class [FriendInfo](#)
Used to store info about a friend's online state and in which room he/she is.

9.19 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/PhotonNetwork/GizmoType.cs File Reference ↩

Classes

- class [ExitGames.Client.GUI.GizmoTypeDrawer](#)

Namespaces

- namespace [ExitGames.Client.GUI](#)

Enumerations

- enum [ExitGames.Client.GUI.GizmoType](#) { [ExitGames.Client.GUI.GizmoType.WireSphere](#), [ExitGames.Client.GUI.GizmoType.Sphere](#), [ExitGames.Client.GUI.GizmoType.WireCube](#), [ExitGames.Client.GUI.GizmoType.Cube](#) }

9.20 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/↵ Plugins/PhotonNetwork/LoadbalancingPeer.cs File Reference

Classes

- class **ExitGames.Client.Photon.LoadbalancingPeer**
Internally used by PUN, a LoadbalancingPeer provides the operations and enum definitions needed to use the [Photon Loadbalancing](#) server (or the [Photon Cloud](#)).
- class [ExitGames.Client.Photon.LoadbalancingPeer.EnterRoomParams](#)
- class [ExitGames.Client.Photon.LoadbalancingPeer.OpJoinRandomRoomParams](#)
- class [ExitGames.Client.Photon.ErrorCode](#)
[ErrorCode](#) defines the default codes associated with [Photon](#) client/server communication.
- class [ExitGames.Client.Photon.ActorProperties](#)
Class for constants.
- class [ExitGames.Client.Photon.GamePropertyKey](#)
Class for constants.
- class [ExitGames.Client.Photon.EventCode](#)
Class for constants.
- class [ExitGames.Client.Photon.ParameterCode](#)
Class for constants.
- class [ExitGames.Client.Photon.OperationCode](#)
Class for constants.
- class [RaiseEventOptions](#)
Aggregates several less-often used options for operation RaiseEvent. See field descriptions for usage details.
- class [TypedLobby](#)
Refers to a specific lobby (and type) on the server.
- class [TypedLobbyInfo](#)
- class [AuthenticationValues](#)
Container for user authentication in [Photon](#).

Namespaces

- namespace [ExitGames.Client.Photon](#)

Typedefs

- using [Hashtable](#) = ExitGames.Client.Photon.Hashtable

Enumerations

- enum [ExitGames.Client.Photon.JoinMode](#) : byte { [ExitGames.Client.Photon.JoinMode.Default](#) = 0, [ExitGames.Client.Photon.JoinMode.CreateIfNotExists](#) = 1, [ExitGames.Client.Photon.JoinMode.JoinOrRejoin](#) = 2, [ExitGames.Client.Photon.JoinMode.RejoinOnly](#) = 3 }
Defines possible values for OpJoinRoom and OpJoinOrCreate.
- enum [ExitGames.Client.Photon.MatchmakingMode](#) : byte { [ExitGames.Client.Photon.MatchmakingMode.FillRoom](#) = 0, [ExitGames.Client.Photon.MatchmakingMode.SerialMatching](#) = 1, [ExitGames.Client.Photon.MatchmakingMode.RandomMatching](#) = 2 }
Options for matchmaking rules for OpJoinRandom.

- enum [ExitGames.Client.Photon.ReceiverGroup](#) : byte { [ExitGames.Client.Photon.ReceiverGroup.Others](#) = 0, [ExitGames.Client.Photon.ReceiverGroup.All](#) = 1, [ExitGames.Client.Photon.ReceiverGroup.MasterClient](#) = 2 }

Lite - OpRaiseEvent lets you chose which actors in the room should receive events.

- enum [ExitGames.Client.Photon.EventCaching](#) : byte { [ExitGames.Client.Photon.EventCaching.DoNotCache](#) = 0, [ExitGames.Client.Photon.EventCaching.MergeCache](#) = 1, [ExitGames.Client.Photon.EventCaching.ReplaceCache](#) = 2, [ExitGames.Client.Photon.EventCaching.RemoveCache](#) = 3, [ExitGames.Client.Photon.EventCaching.AddToRoomCache](#) = 4, [ExitGames.Client.Photon.EventCaching.AddToRoomCacheGlobal](#) = 5, [ExitGames.Client.Photon.EventCaching.RemoveFromRoomCache](#) = 6, [ExitGames.Client.Photon.EventCaching.RemoveFromRoomCacheForActorsLeft](#) = 7, [ExitGames.Client.Photon.EventCaching.SliceIncreaseIndex](#) = 10, [ExitGames.Client.Photon.EventCaching.SliceSetIndex](#) = 11, [ExitGames.Client.Photon.EventCaching.SlicePurgeIndex](#) = 12, [ExitGames.Client.Photon.EventCaching.SlicePurgeUpToIndex](#) = 13 }

Lite - OpRaiseEvent allows you to cache events and automatically send them to joining players in a room.

- enum [ExitGames.Client.Photon.PropertyTypeFlag](#) : byte { [ExitGames.Client.Photon.PropertyTypeFlag.None](#) = 0x00, [ExitGames.Client.Photon.PropertyTypeFlag.Game](#) = 0x01, [ExitGames.Client.Photon.PropertyTypeFlag.Actor](#) = 0x02, [ExitGames.Client.Photon.PropertyTypeFlag.GameAndActor](#) = Game | Actor }

Flags for "types of properties", being used as filter in OpGetProperties.

- enum [LobbyType](#) : byte { [LobbyType.Default](#) = 0, [LobbyType.SqlLobby](#) = 2, [LobbyType.AsyncRandomLobby](#) = 3 }

Options of lobby types available.

- enum [CustomAuthenticationType](#) : byte { [CustomAuthenticationType.Custom](#) = 0, [CustomAuthenticationType.Steam](#) = 1, [CustomAuthenticationType.Facebook](#) = 2, [CustomAuthenticationType.None](#) = byte.MaxValue }

Options for optional "Custom Authentication" services used with [Photon](#).

9.20.1 Typedef Documentation

9.20.1.1 using Hashtable = ExitGames.Client.Photon.Hashtable

9.20.2 Enumeration Type Documentation

9.20.2.1 enum CustomAuthenticationType : byte [strong]

Options for optional "Custom Authentication" services used with [Photon](#).

Used by OpAuthenticate after connecting to [Photon](#).

Enumerator

Custom Use a custom authentication service. Currently the only implemented option.

Steam Authenticates users by their Steam Account. Set auth values accordingly!

Facebook Authenticates users by their Facebook Account. Set auth values accordingly!

None Disables custom authentication. Same as not providing any [AuthenticationValues](#) for connect (more precisely for: OpAuthenticate).

9.20.2.2 enum LobbyType : byte [strong]

Options of lobby types available.

Lobby types might be implemented in certain [Photon](#) versions and won't be available on older servers.

Enumerator

Default This lobby is used unless another is defined by game or JoinRandom. Room-lists will be sent and JoinRandomRoom can filter by matching properties.

SqlLobby This lobby type lists rooms like Default but JoinRandom has a parameter for SQL-like "where" clauses for filtering. This allows bigger, less, or and and combinations.

AsyncRandomLobby This lobby does not send lists of games. It is only used for OpJoinRandomRoom. It keeps rooms available for a while when there are only inactive users left.

9.21 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/PhotonNetwork/NetworkingPeer.cs File Reference ↩

Classes

- class **NetworkingPeer**
Implements [Photon](#) LoadBalancing used in PUN.

Typedefs

- using [Hashtable](#) = ExitGames.Client.Photon.Hashtable

9.21.1 Typedef Documentation

9.21.1.1 using Hashtable = ExitGames.Client.Photon.Hashtable

9.22 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/PhotonNetwork/PhotonClasses.cs File Reference ↩

Wraps up smaller classes that don't need their own file.

Classes

- interface [IPunObservable](#)
Defines the `OnPhotonSerializeView` method to make it easy to implement correctly for observable scripts.
- interface [IPunCallbacks](#)
This interface is used as definition of all callback methods of PUN, except `OnPhotonSerializeView`.
- interface [IPunPrefabPool](#)
Defines all the methods that a Object Pool must implement, so that PUN can use it.
- class [Photon.MonoBehaviour](#)
This class adds the property `photonView`, while logging a warning when your game still uses the `networkView`.
- class [Photon.PunBehaviour](#)
This class provides a `.photonView` and all callbacks/events that PUN can call.
- class [PhotonMessageInfo](#)
Container class for info about a particular message, RPC or update.
- class [RoomOptions](#)
Wraps up common room properties needed when you create rooms.
- class **PunEvent**
Defines [Photon](#) event-codes as used by PUN.
- class [PhotonStream](#)
This container is used in `OnPhotonSerializeView()` to either provide incoming data of a [PhotonView](#) or for you to provide it.
- class [HelpURL](#)
Empty implementation of the upcoming [HelpURL](#) of Unity 5.1.
- class [UnityEngine.SceneManagement.SceneManager](#)
Minimal implementation of the [SceneManager](#) for older Unity, up to v5.2.
- class [SceneManagerHelper](#)
- class [WebRpcResponse](#)
Reads an operation response of a `WebRpc` and provides convenient access to most common values.

Namespaces

- namespace [Photon](#)
- namespace [UnityEngine.SceneManagement](#)

Typedefs

- using [Hashtable](#) = `ExitGames.Client.Photon.Hashtable`
- using [Photon.Hashtable](#) = `ExitGames.Client.Photon.Hashtable`

9.22.1 Detailed Description

Wraps up smaller classes that don't need their own file.

9.22.2 Typedef Documentation

9.22.2.1 using Hashtable = ExitGames.Client.Photon.Hashtable

9.23 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/PhotonNetwork/PhotonHandler.cs File Reference

Classes

- class **PhotonHandler**
Internal MonoBehaviour that allows [Photon](#) to run an Update loop.

Typedefs

- using [Debug](#) = UnityEngine.Debug
- using [Hashtable](#) = ExitGames.Client.Photon.Hashtable

9.23.1 Typedef Documentation

9.23.1.1 using Debug = UnityEngine.Debug

9.23.1.2 using Hashtable = ExitGames.Client.Photon.Hashtable

9.24 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/PhotonNetwork/PhotonLagSimulationGui.cs File Reference

Part of the [Optional GUI](#).

Classes

- class [PhotonLagSimulationGui](#)
This MonoBehaviour is a basic GUI for the [Photon](#) client's network-simulation feature.

9.24.1 Detailed Description

Part of the [Optional GUI](#).

9.25 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/PhotonNetwork/PhotonNetwork.cs File Reference

Classes

- class [PhotonNetwork](#)
The main class to use the [PhotonNetwork](#) plugin.

Typedefs

- using [Debug](#) = UnityEngine.Debug
- using [Hashtable](#) = ExitGames.Client.Photon.Hashtable

9.25.1 Typedef Documentation

9.25.1.1 using [Debug](#) = UnityEngine.Debug

9.25.1.2 using [Hashtable](#) = ExitGames.Client.Photon.Hashtable

9.26 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/PhotonNetwork/PhotonPlayer.cs File Reference

Classes

- class [PhotonPlayer](#)
Summarizes a "player" within a room, identified (in that room) by actorID.

Typedefs

- using [Hashtable](#) = ExitGames.Client.Photon.Hashtable

9.26.1 Typedef Documentation

9.26.1.1 using [Hashtable](#) = ExitGames.Client.Photon.Hashtable

9.27 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/PhotonNetwork/PhotonStatsGui.cs File Reference

Part of the [Optional GUI](#).

Classes

- class [PhotonStatsGui](#)
Basic GUI to show traffic and health statistics of the connection to [Photon](#), toggled by shift+tab.

9.27.1 Detailed Description

Part of the [Optional GUI](#).

9.28 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/PhotonNetwork/PhotonStreamQueue.cs File Reference ↩

Classes

- class [PhotonStreamQueue](#)

The [PhotonStreamQueue](#) helps you poll object states at higher frequencies than what [PhotonNetwork.sendRate](#) dictates and then sends all those states at once when [Serialize\(\)](#) is called.

9.29 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/PhotonNetwork/PhotonView.cs File Reference ↩

Classes

- class [PhotonView](#)

PUN's [NetworkView](#) replacement class for networking.

Enumerations

- enum [ViewSynchronization](#) { [ViewSynchronization.Off](#), [ViewSynchronization.ReliableDeltaCompressed](#), [ViewSynchronization.Unreliable](#), [ViewSynchronization.UnreliableOnChange](#) }
- enum [OnSerializeTransform](#) { [OnSerializeTransform.OnlyPosition](#), [OnSerializeTransform.OnlyRotation](#), [OnSerializeTransform.OnlyScale](#), [OnSerializeTransform.PositionAndRotation](#), [OnSerializeTransform.All](#) }
- enum [OnSerializeRigidBody](#) { [OnSerializeRigidBody.OnlyVelocity](#), [OnSerializeRigidBody.OnlyAngularVelocity](#), [OnSerializeRigidBody.All](#) }
- enum [OwnershipOption](#) { [OwnershipOption.Fixed](#), [OwnershipOption.Takeover](#), [OwnershipOption.Request](#) }

Options to define how Ownership Transfer is handled per [PhotonView](#).

9.29.1 Enumeration Type Documentation

9.29.1.1 enum [OnSerializeRigidBody](#) [strong]

Enumerator

OnlyVelocity

OnlyAngularVelocity

All

9.29.1.2 enum [OnSerializeTransform](#) [strong]

Enumerator

OnlyPosition

OnlyRotation

OnlyScale

PositionAndRotation

All

9.29.1.3 enum OwnershipOption [strong]

Options to define how Ownership Transfer is handled per [PhotonView](#).

This setting affects how RequestOwnership and TransferOwnership work at runtime.

Enumerator

Fixed Ownership is fixed. Instantiated objects stick with their creator, scene objects always belong to the Master Client.

Takeover Ownership can be taken away from the current owner who can't object.

Request Ownership can be requested with [PhotonView.RequestOwnership](#) but the current owner has to agree to give up ownership. The current owner has to implement [IPunCallbacks.OnOwnershipRequest](#) to react to the ownership request.

9.29.1.4 enum ViewSynchronization [strong]

Enumerator

Off

ReliableDeltaCompressed

Unreliable

UnreliableOnChange

9.30 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/PhotonNetwork/PingCloudRegions.cs File Reference ↩

Classes

- class [PingMonoEditor](#)
Uses C# Socket class from System.Net.Sockets (as Unity usually does).
- class [PhotonPingManager](#)

Typedefs

- using [Debug](#) = UnityEngine.Debug

9.30.1 Typedef Documentation

9.30.1.1 using Debug = UnityEngine.Debug

9.31 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/PhotonNetwork/Room.cs File Reference ↩

Classes

- class [Room](#)
This class resembles a room that PUN joins (or joined).

9.32 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/PhotonNetwork/RoomInfo.cs File Reference ↩

Classes

- class [RoomInfo](#)

A simplified room with just the info required to list and join, used for the room listing in the lobby.

9.33 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/PhotonNetwork/RPC.cs File Reference ↩

Reimplements a RPC Attribute, as it's no longer in all versions of the [UnityEngine](#) assembly.

Classes

- class [PunRPC](#)

Replacement for RPC attribute with different name. Used to flag methods as remote-callable.

9.33.1 Detailed Description

Reimplements a RPC Attribute, as it's no longer in all versions of the [UnityEngine](#) assembly.

9.34 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/PhotonNetwork/RpcIndexComponent.cs File Reference ↩

Outdated.

9.34.1 Detailed Description

Outdated.

Here to overwrite older files on import.

9.35 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/PhotonNetwork/ServerSettings.cs File Reference ↩

ScriptableObject defining a server setup.

Classes

- class [Region](#)
- class [ServerSettings](#)

Collection of connection-relevant settings, used internally by [PhotonNetwork.ConnectUsingSettings](#).

9.35.1 Detailed Description

ScriptableObject defining a server setup.

An instance is created as **PhotonServerSettings**.

9.36 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/PhotonNetwork/SocketUdp.cs File Reference ↩

Classes

- class **ExitGames.Client.Photon.SocketUdp**

Internal class to encapsulate the network i/o functionality for the realtime library.

Namespaces

- namespace [ExitGames.Client.Photon](#)

9.37 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/PhotonNetwork/SocketWebTcp.cs File Reference ↩

9.38 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/PhotonNetwork/Views/PhotonAnimatorView.cs File Reference ↩

Classes

- class [PhotonAnimatorView](#)
This class helps you to synchronize Mecanim animations Simply add the component to your GameObject and make sure that the [PhotonAnimatorView](#) is added to the list of observed components
- class [PhotonAnimatorView.SynchronizedParameter](#)
- class [PhotonAnimatorView.SynchronizedLayer](#)

9.39 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/PhotonNetwork/Views/PhotonRigidbody2DView.cs File Reference ↩

Classes

- class [PhotonRigidbody2DView](#)

This class helps you to synchronize the velocities of a 2d physics Rigidbody.

9.40 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/↔ Plugins/PhotonNetwork/Views/PhotonRigidbodyView.cs File Reference

Classes

- class [PhotonRigidbodyView](#)

This class helps you to synchronize the velocities of a physics Rigidbody.

9.41 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/↔ Plugins/PhotonNetwork/Views/PhotonTransformView.cs File Reference

Classes

- class [PhotonTransformView](#)

This class helps you to synchronize position, rotation and scale of a GameObject.

9.42 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/↔ Plugins/PhotonNetwork/Views/PhotonTransformViewPositionControl.cs File Reference

Classes

- class [PhotonTransformViewPositionControl](#)

9.43 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/↔ Plugins/PhotonNetwork/Views/PhotonTransformViewPositionModel.cs File Reference

Classes

- class [PhotonTransformViewPositionModel](#)

9.44 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/↔ Plugins/PhotonNetwork/Views/PhotonTransformViewRotationControl.cs File Reference

Classes

- class [PhotonTransformViewRotationControl](#)

9.45 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/PhotonNetwork/Views/PhotonTransformViewRotationModel.cs File Reference

Classes

- class [PhotonTransformViewRotationModel](#)

9.46 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/PhotonNetwork/Views/PhotonTransformViewScaleControl.cs File Reference

Classes

- class [PhotonTransformViewScaleControl](#)

9.47 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/Plugins/PhotonNetwork/Views/PhotonTransformViewScaleModel.cs File Reference

Classes

- class [PhotonTransformViewScaleModel](#)

9.48 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/UtilityScripts/ConnectAndJoinRandom.cs File Reference

Classes

- class [ConnectAndJoinRandom](#)

This script automatically connects to [Photon](#) (using the settings file), tries to join a random room and creates one if none was found (which is ok).

9.49 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/UtilityScripts/HighlightOwnedGameObj.cs File Reference

Classes

- class [HighlightOwnedGameObj](#)

9.50 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/UtilityScripts/InputToEvent.cs File Reference ↩↪

Classes

- class [InputToEvent](#)
Utility component to forward mouse or touch input to clicked gameobjects.

9.51 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/UtilityScripts/InRoomChat.cs File Reference ↩↪

Classes

- class [InRoomChat](#)

9.52 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/UtilityScripts/InRoomRoundTimer.cs File Reference ↩↪

Classes

- class [InRoomRoundTimer](#)
Simple script that uses a property to sync a start time for a multiplayer game.

9.53 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/UtilityScripts/ManualPhotonViewAllocator.cs File Reference ↩↪

Classes

- class [ManualPhotonViewAllocator](#)

9.54 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/UtilityScripts/MoveByKeys.cs File Reference ↩↪

Classes

- class [MoveByKeys](#)
Very basic component to move a GameObject by WASD and Space.

9.55 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/UtilityScripts/OnAwakeUsePhotonView.cs File Reference ↩

Classes

- class [OnAwakeUsePhotonView](#)

9.56 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/UtilityScripts/OnClickDestroy.cs File Reference ↩

Classes

- class [OnClickDestroy](#)
Implements OnClick to destroy the GameObject it's attached to.

9.57 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/UtilityScripts/OnClickInstantiate.cs File Reference ↩

Classes

- class [OnClickInstantiate](#)

9.58 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/UtilityScripts/OnClickLoadSomething.cs File Reference ↩

Classes

- class [OnClickLoadSomething](#)
This component makes it easy to switch scenes or open webpages on click.

9.59 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/UtilityScripts/OnJoinedInstantiate.cs File Reference ↩

Classes

- class [OnJoinedInstantiate](#)

9.60 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/UtilityScripts/OnStartDelete.cs File Reference ↩

Classes

- class [OnStartDelete](#)
This component will destroy the GameObject it is attached to (in Start()).

9.61 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/UtilityScripts/PickupItem.cs File Reference ↩

Classes

- class [PickupItem](#)
Makes a scene object pickup-able.

Typedefs

- using [Hashtable](#) = ExitGames.Client.Photon.Hashtable

9.61.1 Typedef Documentation

9.61.1.1 using [Hashtable](#) = ExitGames.Client.Photon.Hashtable

9.62 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/UtilityScripts/PickupItemSimple.cs File Reference ↩

Classes

- class [PickupItemSimple](#)
Makes a scene object pickup-able.

9.63 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/UtilityScripts/PickupItemSyncer.cs File Reference ↩

Classes

- class [PickupItemSyncer](#)
Finds out which PickupItems are not spawned at the moment and send this to new players.

Typedefs

- using [Hashtable](#) = ExitGames.Client.Photon.Hashtable

9.63.1 Typedef Documentation

9.63.1.1 using [Hashtable](#) = ExitGames.Client.Photon.Hashtable

9.64 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/UtilityScripts/PointedAtGameObjectInfo.cs File Reference ↩

Classes

- class [PointedAtGameObjectInfo](#)

9.65 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/ UtilityScripts/PunPlayerScores.cs File Reference ↩

Classes

- class [PunPlayerScores](#)
- class [ScoreExtensions](#)

Typedefs

- using [Hashtable](#) = ExitGames.Client.Photon.Hashtable

9.65.1 Typedef Documentation

9.65.1.1 using [Hashtable](#) = ExitGames.Client.Photon.Hashtable

9.66 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/ UtilityScripts/PunTeams.cs File Reference ↩

Classes

- class [PunTeams](#)
Implements teams in a room/game with help of player properties.
- class [TeamExtensions](#)
Extension used for [PunTeams](#) and [PhotonPlayer](#) class. Wraps access to the player's custom property.

Typedefs

- using [Hashtable](#) = ExitGames.Client.Photon.Hashtable

9.66.1 Typedef Documentation

9.66.1.1 using [Hashtable](#) = ExitGames.Client.Photon.Hashtable

9.67 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/ UtilityScripts/QuitOnEscapeOrBack.cs File Reference ↩

Classes

- class [QuitOnEscapeOrBack](#)

9.68 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/UtilityScripts/ServerTime.cs File Reference ↩

Classes

- class [ServerTime](#)

9.69 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/UtilityScripts/ShowInfoOfPlayer.cs File Reference ↩

Classes

- class [ShowInfoOfPlayer](#)
Can be attached to a `GameObject` to show info about the owner of the [PhotonView](#).

9.70 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/UtilityScripts/ShowStatusWhenConnecting.cs File Reference ↩

Classes

- class [ShowStatusWhenConnecting](#)

9.71 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/UtilityScripts/SmoothSyncMovement.cs File Reference ↩

Classes

- class [SmoothSyncMovement](#)

9.72 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/UtilityScripts/SupportLogger.cs File Reference ↩

Classes

- class [SupportLogger](#)
- class [SupportLogging](#)

9.73 C:/Users/Chris/Documents/GitHub/Team-Game-Project/Assets/Photon Unity Networking/UtilityScripts/TimeKeeper.cs File Reference ↩

Classes

- class [ExitGames.Client.DemoParticle.TimeKeeper](#)
A utility class that turns it's `ShouldExecute` property to true after a set interval time has passed.

Namespaces

- namespace [ExitGames.Client.DemoParticle](#)

Index

- AccountService, [33](#)
 - AccountService, [34](#)
 - AppId, [34](#)
 - CloudWeb, [33](#)
 - Message, [34](#)
 - Origin, [33](#)
 - Playmaker, [33](#)
 - Pun, [33](#)
 - RegisterByEmail, [34](#)
 - RegisterByEmailAsync, [34](#)
 - ReturnCode, [34](#)
 - ServerWeb, [33](#)
 - Validator, [34](#)
- ActiveSceneBuildIndex
 - SceneManagerHelper, [203](#)
- ActiveSceneName
 - SceneManagerHelper, [203](#)
- Actor
 - ExitGames::Client::Photon, [30](#)
- ActorList
 - ExitGames::Client::Photon::ParameterCode, [83](#)
- ActorNr
 - ExitGames::Client::Photon::ParameterCode, [83](#)
- Add
 - ExitGames::Client::Photon::ParameterCode, [83](#)
- AddAuthParameter
 - AuthenticationValues, [37](#)
- AddButton
 - PhotonGUI, [97](#)
- AddLine
 - InRoomChat, [56](#)
- AddScore
 - ScoreExtensions, [204](#)
- AddToRoomCache
 - ExitGames::Client::Photon, [29](#)
- AddToRoomCacheGlobal
 - ExitGames::Client::Photon, [29](#)
- Address
 - ExitGames::Client::Photon::ParameterCode, [83](#)
- AlignBottom
 - InRoomChat, [56](#)
- All
 - ExitGames::Client::Photon, [30](#)
 - PhotonView.cs, [232](#)
 - Public API, [24](#)
- AllBuffered
 - Public API, [24](#)
- AllBufferedViaServer
 - Public API, [25](#)
- allProperties
 - PhotonPlayer, [142](#)
- AllViaServer
 - Public API, [25](#)
- AllocateManualPhotonView
 - ManualPhotonViewAllocator, [69](#)
- AllocateSceneViewID
 - PhotonNetwork, [106](#)
- AllocateViewID
 - PhotonNetwork, [106](#)
- AlmostEquals
 - Extensions, [47](#)
- AlreadyMatched
 - ExitGames::Client::Photon::ErrorCode, [41](#)
- AlreadyRegisteredInfo
 - PunWizardText, [188](#)
- AppId
 - ServerSettings, [206](#)
- AppId
 - AccountService, [34](#)
- AppStats
 - ExitGames::Client::Photon::EventCode, [45](#)
- AppVersion
 - ExitGames::Client::Photon::ParameterCode, [83](#)
- ApplicationId
 - ExitGames::Client::Photon::ParameterCode, [83](#)
- AppliedToSettingsInfo
 - PunWizardText, [188](#)
- asia
 - Enums.cs, [224](#)
- AskForPickupItemSpawnTimes
 - PickupItemSyncer, [172](#)
- AsyncRandomLobby
 - LoadbalancingPeer.cs, [228](#)
- Attempts
 - PhotonPingManager, [138](#)
- au
 - Enums.cs, [224](#)
- AuthGetParameters
 - AuthenticationValues, [37](#)
- AuthPostData
 - AuthenticationValues, [37](#)
- AuthType
 - AuthenticationValues, [38](#)
- AuthValues
 - PhotonNetwork, [129](#)
- Authenticate
 - ExitGames::Client::Photon::OperationCode, [77](#)
- Authenticated

- Public API, [20](#)
- Authenticating
 - Public API, [20](#)
- AuthenticationTicketExpired
 - ExitGames::Client::Photon::ErrorCode, [41](#)
 - Public API, [19](#)
- AuthenticationValues, [35](#)
 - AddAuthParameter, [37](#)
 - AuthGetParameters, [37](#)
 - AuthPostData, [37](#)
 - AuthType, [38](#)
 - AuthenticationValues, [36](#)
 - SetAuthPostData, [37](#)
 - ToString, [37](#)
 - Token, [38](#)
 - UserId, [38](#)
- autoCleanUp
 - Room, [196](#)
- autoCleanUpField
 - RoomInfo, [199](#)
- autoCleanUpPlayerObjects
 - PhotonNetwork, [129](#)
- AutoConnect
 - ConnectAndJoinRandom, [39](#)
- autoJoinLobby
 - PhotonNetwork, [129](#)
- automaticallySyncScene
 - PhotonNetwork, [130](#)
- Awake
 - SmoothSyncMovement, [209](#)
- AzureLocalNodeId
 - ExitGames::Client::Photon::ParameterCode, [83](#)
- AzureMasterNodeId
 - ExitGames::Client::Photon::ParameterCode, [84](#)
- AzureNodeIdInfo
 - ExitGames::Client::Photon::EventCode, [45](#)
 - ExitGames::Client::Photon::ParameterCode, [84](#)
- BackgroundTimeout
 - PhotonNetwork, [127](#)
- BestRegion
 - PhotonPingManager, [138](#)
 - ServerSettings, [205](#)
- blue
 - PunTeams, [185](#)
- Bool
 - PhotonAnimatorView, [91](#)
- Broadcast
 - ExitGames::Client::Photon::ParameterCode, [84](#)
- buttonsOn
 - PhotonStatsGui, [146](#)
- C:/Users/Chris/Documents/GitHub/Team-Game-↵
 - Project/Assets/Photon Unity Networking/↵
 - Editor/PhotonNetwork/AccountService.cs, [219](#)
- C:/Users/Chris/Documents/GitHub/Team-Game-↵
 - Project/Assets/Photon Unity Networking/↵
 - Editor/PhotonNetwork/PhotonConverter.cs, [219](#)
 - Project/Assets/Photon Unity Networking/↵
 - Editor/PhotonNetwork/PhotonEditor.cs, [219](#)
 - Project/Assets/Photon Unity Networking/↵
 - Editor/PhotonNetwork/PhotonGUI.cs, [219](#)
 - Project/Assets/Photon Unity Networking/↵
 - Editor/PhotonNetwork/PhotonViewHandler.↵
 cs, [220](#)
 - Project/Assets/Photon Unity Networking/↵
 - Editor/PhotonNetwork/PhotonViewInspector.↵
 cs, [220](#)
 - Project/Assets/Photon Unity Networking/↵
 - Editor/PhotonNetwork/PhotonViewPrefab↵
 Apply.cs, [220](#)
 - Project/Assets/Photon Unity Networking/↵
 - Editor/PhotonNetwork/PunSceneSettings.cs, [220](#)
 - Project/Assets/Photon Unity Networking/↵
 - Editor/PhotonNetwork/ReorderableList↵
 Resources.cs, [220](#)
 - Project/Assets/Photon Unity Networking/↵
 - Editor/PhotonNetwork/ServerSettings↵
 Inspector.cs, [221](#)
 - Project/Assets/Photon Unity Networking/↵
 - Editor/PhotonNetwork/Views/Photon↵
 AnimatorViewEditor.cs, [221](#)
 - Project/Assets/Photon Unity Networking/↵
 - Editor/PhotonNetwork/Views/Photon↵
 Rigidbody2DViewEditor.cs, [221](#)
 - Project/Assets/Photon Unity Networking/↵
 - Editor/PhotonNetwork/Views/Photon↵
 RigidbodyViewEditor.cs, [221](#)
 - Project/Assets/Photon Unity Networking/↵
 - Editor/PhotonNetwork/Views/Photon↵
 TransformViewEditor.cs, [221](#)
 - Project/Assets/Photon Unity Networking/↵
 - Plugins/PhotonNetwork/CustomTypes.cs, [222](#)
 - Project/Assets/Photon Unity Networking/↵
 - Plugins/PhotonNetwork/Enums.cs, [222](#)
 - Project/Assets/Photon Unity Networking/↵
 - Plugins/PhotonNetwork/Extensions.cs, [225](#)
- C:/Users/Chris/Documents/GitHub/Team-Game-↵
 - Project/Assets/Photon Unity Networking/↵

- Project/Assets/Photon Unity Networking/↵
Plugins/PhotonNetwork/FriendInfo.cs, [225](#)
- C:/Users/Chris/Documents/GitHub/Team-Game-↵
Project/Assets/Photon Unity Networking/↵
Plugins/PhotonNetwork/GizmoType.cs, [225](#)
- C:/Users/Chris/Documents/GitHub/Team-Game-↵
Project/Assets/Photon Unity Networking/↵
Plugins/PhotonNetwork/Loadbalancing↵
Peer.cs, [226](#)
- C:/Users/Chris/Documents/GitHub/Team-Game-↵
Project/Assets/Photon Unity Networking/↵
Plugins/PhotonNetwork/NetworkingPeer.cs, [228](#)
- C:/Users/Chris/Documents/GitHub/Team-Game-↵
Project/Assets/Photon Unity Networking/↵
Plugins/PhotonNetwork/PhotonClasses.cs, [228](#)
- C:/Users/Chris/Documents/GitHub/Team-Game-↵
Project/Assets/Photon Unity Networking/↵
Plugins/PhotonNetwork/PhotonHandler.cs, [230](#)
- C:/Users/Chris/Documents/GitHub/Team-Game-↵
Project/Assets/Photon Unity Networking/↵
Plugins/PhotonNetwork/PhotonLagSimulation↵
Gui.cs, [230](#)
- C:/Users/Chris/Documents/GitHub/Team-Game-↵
Project/Assets/Photon Unity Networking/↵
Plugins/PhotonNetwork/PhotonNetwork.cs, [230](#)
- C:/Users/Chris/Documents/GitHub/Team-Game-↵
Project/Assets/Photon Unity Networking/↵
Plugins/PhotonNetwork/PhotonPlayer.cs, [231](#)
- C:/Users/Chris/Documents/GitHub/Team-Game-↵
Project/Assets/Photon Unity Networking/↵
Plugins/PhotonNetwork/PhotonStatsGui.cs, [231](#)
- C:/Users/Chris/Documents/GitHub/Team-Game-↵
Project/Assets/Photon Unity Networking/↵
Plugins/PhotonNetwork/PhotonStream↵
Queue.cs, [232](#)
- C:/Users/Chris/Documents/GitHub/Team-Game-↵
Project/Assets/Photon Unity Networking/↵
Plugins/PhotonNetwork/PhotonView.cs, [232](#)
- C:/Users/Chris/Documents/GitHub/Team-Game-↵
Project/Assets/Photon Unity Networking/↵
Plugins/PhotonNetwork/PingCloudRegions.↵
cs, [233](#)
- C:/Users/Chris/Documents/GitHub/Team-Game-↵
Project/Assets/Photon Unity Networking/↵
Plugins/PhotonNetwork/RPC.cs, [234](#)
- C:/Users/Chris/Documents/GitHub/Team-Game-↵
Project/Assets/Photon Unity Networking/↵
Plugins/PhotonNetwork/Room.cs, [233](#)
- C:/Users/Chris/Documents/GitHub/Team-Game-↵
Project/Assets/Photon Unity Networking/↵
Plugins/PhotonNetwork/RoomInfo.cs, [234](#)
- C:/Users/Chris/Documents/GitHub/Team-Game-↵
Project/Assets/Photon Unity Networking/↵
Plugins/PhotonNetwork/RpcIndexComponent.↵
cs, [234](#)
- C:/Users/Chris/Documents/GitHub/Team-Game-↵
Project/Assets/Photon Unity Networking/↵
Plugins/PhotonNetwork/ServerSettings.cs, [234](#)
- C:/Users/Chris/Documents/GitHub/Team-Game-↵
Project/Assets/Photon Unity Networking/↵
Plugins/PhotonNetwork/SocketUdp.cs, [235](#)
- C:/Users/Chris/Documents/GitHub/Team-Game-↵
Project/Assets/Photon Unity Networking/↵
Plugins/PhotonNetwork/SocketWebTcp.cs, [235](#)
- C:/Users/Chris/Documents/GitHub/Team-Game-↵
Project/Assets/Photon Unity Networking/↵
Plugins/PhotonNetwork/Views/Photon↵
AnimatorView.cs, [235](#)
- C:/Users/Chris/Documents/GitHub/Team-Game-↵
Project/Assets/Photon Unity Networking/↵
Plugins/PhotonNetwork/Views/Photon↵
Rigidbody2DView.cs, [235](#)
- C:/Users/Chris/Documents/GitHub/Team-Game-↵
Project/Assets/Photon Unity Networking/↵
Plugins/PhotonNetwork/Views/Photon↵
RigidbodyView.cs, [236](#)
- C:/Users/Chris/Documents/GitHub/Team-Game-↵
Project/Assets/Photon Unity Networking/↵
Plugins/PhotonNetwork/Views/Photon↵
TransformView.cs, [236](#)
- C:/Users/Chris/Documents/GitHub/Team-Game-↵
Project/Assets/Photon Unity Networking/↵
Plugins/PhotonNetwork/Views/Photon↵
TransformViewPositionControl.cs, [236](#)
- C:/Users/Chris/Documents/GitHub/Team-Game-↵
Project/Assets/Photon Unity Networking/↵
Plugins/PhotonNetwork/Views/Photon↵
TransformViewPositionModel.cs, [236](#)
- C:/Users/Chris/Documents/GitHub/Team-Game-↵
Project/Assets/Photon Unity Networking/↵
Plugins/PhotonNetwork/Views/Photon↵
TransformViewRotationControl.cs, [236](#)
- C:/Users/Chris/Documents/GitHub/Team-Game-↵
Project/Assets/Photon Unity Networking/↵
Plugins/PhotonNetwork/Views/Photon↵
TransformViewRotationModel.cs, [237](#)
- C:/Users/Chris/Documents/GitHub/Team-Game-↵
Project/Assets/Photon Unity Networking/↵
Plugins/PhotonNetwork/Views/Photon↵
TransformViewScaleControl.cs, [237](#)
- C:/Users/Chris/Documents/GitHub/Team-Game-↵
Project/Assets/Photon Unity Networking/↵
Plugins/PhotonNetwork/Views/Photon↵
TransformViewScaleModel.cs, [237](#)
- C:/Users/Chris/Documents/GitHub/Team-Game-↵
Project/Assets/Photon Unity Networking/↵
UtilityScripts/ConnectAndJoinRandom.cs, [237](#)
- C:/Users/Chris/Documents/GitHub/Team-Game-↵

- Project/Assets/Photon Unity Networking/↵
UtilityScripts/HighlightOwnedGameObj.cs, [237](#)
- C:/Users/Chris/Documents/GitHub/Team-Game-↵
Project/Assets/Photon Unity Networking/↵
UtilityScripts/InRoomChat.cs, [238](#)
- C:/Users/Chris/Documents/GitHub/Team-Game-↵
Project/Assets/Photon Unity Networking/↵
UtilityScripts/InRoomRoundTimer.cs, [238](#)
- C:/Users/Chris/Documents/GitHub/Team-Game-↵
Project/Assets/Photon Unity Networking/↵
UtilityScripts/InputToEvent.cs, [238](#)
- C:/Users/Chris/Documents/GitHub/Team-Game-↵
Project/Assets/Photon Unity Networking/↵
UtilityScripts/ManualPhotonViewAllocator.cs, [238](#)
- C:/Users/Chris/Documents/GitHub/Team-Game-↵
Project/Assets/Photon Unity Networking/↵
UtilityScripts/MoveByKeys.cs, [238](#)
- C:/Users/Chris/Documents/GitHub/Team-Game-↵
Project/Assets/Photon Unity Networking/↵
UtilityScripts/OnAwakeUsePhotonView.cs, [239](#)
- C:/Users/Chris/Documents/GitHub/Team-Game-↵
Project/Assets/Photon Unity Networking/↵
UtilityScripts/OnClickDestroy.cs, [239](#)
- C:/Users/Chris/Documents/GitHub/Team-Game-↵
Project/Assets/Photon Unity Networking/↵
UtilityScripts/OnClickInstantiate.cs, [239](#)
- C:/Users/Chris/Documents/GitHub/Team-Game-↵
Project/Assets/Photon Unity Networking/↵
UtilityScripts/OnClickLoadSomething.cs, [239](#)
- C:/Users/Chris/Documents/GitHub/Team-Game-↵
Project/Assets/Photon Unity Networking/↵
UtilityScripts/OnJoinedInstantiate.cs, [239](#)
- C:/Users/Chris/Documents/GitHub/Team-Game-↵
Project/Assets/Photon Unity Networking/↵
UtilityScripts/OnStartDelete.cs, [239](#)
- C:/Users/Chris/Documents/GitHub/Team-Game-↵
Project/Assets/Photon Unity Networking/↵
UtilityScripts/PickupItem.cs, [240](#)
- C:/Users/Chris/Documents/GitHub/Team-Game-↵
Project/Assets/Photon Unity Networking/↵
UtilityScripts/PickupItemSimple.cs, [240](#)
- C:/Users/Chris/Documents/GitHub/Team-Game-↵
Project/Assets/Photon Unity Networking/↵
UtilityScripts/PickupItemSyncer.cs, [240](#)
- C:/Users/Chris/Documents/GitHub/Team-Game-↵
Project/Assets/Photon Unity Networking/↵
UtilityScripts/PointedAtGameObjectInfo.cs, [240](#)
- C:/Users/Chris/Documents/GitHub/Team-Game-↵
Project/Assets/Photon Unity Networking/↵
UtilityScripts/PunPlayerScores.cs, [241](#)
- C:/Users/Chris/Documents/GitHub/Team-Game-↵
Project/Assets/Photon Unity Networking/↵
UtilityScripts/PunTeams.cs, [241](#)
- C:/Users/Chris/Documents/GitHub/Team-Game-↵
Project/Assets/Photon Unity Networking/↵
UtilityScripts/QuitOnEscapeOrBack.cs, [241](#)
- C:/Users/Chris/Documents/GitHub/Team-Game-↵
Project/Assets/Photon Unity Networking/↵
UtilityScripts/ServerTime.cs, [242](#)
- C:/Users/Chris/Documents/GitHub/Team-Game-↵
Project/Assets/Photon Unity Networking/↵
UtilityScripts/ShowInfoOfPlayer.cs, [242](#)
- C:/Users/Chris/Documents/GitHub/Team-Game-↵
Project/Assets/Photon Unity Networking/↵
UtilityScripts/ShowStatusWhenConnecting.↵
cs, [242](#)
- C:/Users/Chris/Documents/GitHub/Team-Game-↵
Project/Assets/Photon Unity Networking/↵
UtilityScripts/SmoothSyncMovement.cs, [242](#)
- C:/Users/Chris/Documents/GitHub/Team-Game-↵
Project/Assets/Photon Unity Networking/↵
UtilityScripts/SupportLogger.cs, [242](#)
- C:/Users/Chris/Documents/GitHub/Team-Game-↵
Project/Assets/Photon Unity Networking/↵
UtilityScripts/TimeKeeper.cs, [242](#)
- Cache
 - ExitGames::Client::Photon::ParameterCode, [84](#)
- CacheSendMonoMessageTargets
 - PhotonNetwork, [106](#)
- CacheSliceChanged
 - ExitGames::Client::Photon::EventCode, [45](#)
- CacheSliceIndex
 - ExitGames::Client::Photon::ParameterCode, [84](#)
- CachingOption
 - RaiseEventOptions, [192](#)
- CancelButton
 - PunWizardText, [188](#)
- ChangeGroups
 - ExitGames::Client::Photon::OperationCode, [77](#)
- CharacterSize
 - ShowInfoOfPlayer, [208](#)
- Chat
 - InRoomChat, [56](#)
- ChatRPC
 - InRoomChat, [56](#)
- CheckUserOnJoin
 - ExitGames::Client::Photon::ParameterCode, [84](#)
- CleanupCacheOnLeave
 - ExitGames::Client::Photon::GamePropertyKey, [52](#)
 - ExitGames::Client::Photon::ParameterCode, [84](#)
- cleanupCacheOnLeave
 - RoomOptions, [202](#)
- ClearRpcList
 - PhotonEditor, [95](#)
- ClientAuthenticationData
 - ExitGames::Client::Photon::ParameterCode, [84](#)
- ClientAuthenticationParams
 - ExitGames::Client::Photon::ParameterCode, [84](#)
- ClientAuthenticationType
 - ExitGames::Client::Photon::ParameterCode, [84](#)
- CloseConnection
 - PhotonNetwork, [107](#)

- CloseWindowButton
 - PunWizardText, 188
- CloudRegionCode
 - Enums.cs, 223
- CloudRegionFlag
 - Enums.cs, 224
- CloudWeb
 - AccountService, 33
- Code
 - ExitGames::Client::Photon::ParameterCode, 85
 - Region, 193
- ComparisonPageButton
 - PunWizardText, 188
- ConnectAndJoinRandom, 38
 - AutoConnect, 39
 - OnConnectedToMaster, 39
 - OnFailedToConnectToPhoton, 39
 - OnJoinedLobby, 39
 - OnJoinedRoom, 39
 - OnPhotonRandomJoinFailed, 39
 - Start, 39
 - Update, 39
 - Version, 39
- ConnectToBestCloudServer
 - PhotonNetwork, 107
- ConnectToMaster
 - PhotonNetwork, 108
- ConnectToRegion
 - PhotonNetwork, 108
- ConnectUsingSettings
 - PhotonNetwork, 108
- Connected
 - Enums.cs, 224
- connected
 - PhotonNetwork, 130
- connectedAndReady
 - PhotonNetwork, 130
- ConnectedToGameserver
 - Public API, 20
- ConnectedToMaster
 - Public API, 20
- ConnectedToNameServer
 - Public API, 20
- Connecting
 - Enums.cs, 224
- connecting
 - PhotonNetwork, 130
- ConnectingToGameserver
 - Public API, 20
- ConnectingToMasterserver
 - Public API, 20
- ConnectingToNameServer
 - Public API, 20
- ConnectionInfo
 - PunWizardText, 188
- ConnectionState
 - Enums.cs, 224
- connectionState
 - PhotonNetwork, 130
- connectionStateDetailed
 - PhotonNetwork, 130
- ConnectionTitle
 - PunWizardText, 188
- ContainerBody
 - PhotonGUI, 97
- ContainerHeader
 - PhotonGUI, 97
- ContainerHeaderFoldout
 - PhotonGUI, 97
- ContainerHeaderToggle
 - PhotonGUI, 97
- Contains
 - Extensions, 48
- Continuous
 - PhotonAnimatorView, 91
- ConvertRpcAttribute
 - PhotonConverter, 94
- ConverterLabel
 - PunWizardText, 188
- Count
 - PhotonStream, 149
- countOfPlayers
 - PhotonNetwork, 131
- countOfPlayersInRooms
 - PhotonNetwork, 131
- countOfPlayersOnMaster
 - PhotonNetwork, 131
- countOfRooms
 - PhotonNetwork, 131
- CrcCheckEnabled
 - PhotonNetwork, 131
- CreateGame
 - ExitGames::Client::Photon::OperationCode, 78
- CreateIfNotExists
 - ExitGames::Client::Photon, 29
 - ExitGames::Client::Photon::LoadbalancingPeer::↔
EnterRoomParams, 40
- CreateRoom
 - PhotonNetwork, 109
- CreatorActorNr
 - PhotonView, 165
- Cube
 - ExitGames::Client::GUI, 28
- CurrentLang
 - PhotonEditor, 96
- Custom
 - LoadbalancingPeer.cs, 227
- CustomAuthenticationFailed
 - ExitGames::Client::Photon::ErrorCode, 41
- CustomAuthenticationType
 - LoadbalancingPeer.cs, 227
- CustomEventContent
 - ExitGames::Client::Photon::ParameterCode, 85
- customProperties
 - PhotonPlayer, 142
 - RoomInfo, 199

- customRoomProperties
 - RoomOptions, 201
- customRoomPropertiesForLobby
 - RoomOptions, 201
- Data
 - ExitGames::Client::Photon::ParameterCode, 85
- Debug
 - PhotonHandler.cs, 230
 - PhotonNetwork.cs, 231
 - PhotonViewHandler.cs, 220
 - PingCloudRegions.cs, 233
- DebugMessage
 - WebRpcResponse, 218
- Default
 - ExitGames::Client::Photon, 29
 - LoadbalancingPeer.cs, 228
 - RaiseEventOptions, 192
 - TypedLobby, 216
- DefaultAddButtonStyle
 - PhotonGUI, 97
- DefaultContainerRowStyle
 - PhotonGUI, 97
- DefaultContainerStyle
 - PhotonGUI, 98
- DefaultRemoveButtonStyle
 - PhotonGUI, 98
- DefaultTitleStyle
 - PhotonGUI, 98
- Deserialize
 - PhotonStreamQueue, 151
- DeserializeView
 - PhotonView, 162
- Destroy
 - IPunPrefabPool, 68
 - PhotonNetwork, 110
- DestroyAll
 - PhotonNetwork, 111
- DestroyByRpc
 - OnClickDestroy, 73
- DestroyPlayerObjects
 - PhotonNetwork, 111, 112
- DestroyRpc
 - OnClickDestroy, 73
- DetectPointedAtGameObject
 - InputToEvent, 55
- dir
 - MoveByKeys, 71
- dirFix
 - MoveByKeys, 71
- DisableAutoOpenWizard
 - ServerSettings, 206
- DisableOnOwnObjects
 - ShowInfoOfPlayer, 208
- disableTime
 - MoveByKeys, 71
- Disabled
 - PhotonAnimatorView, 91
 - PhotonTransformViewPositionModel, 156
 - PhotonTransformViewRotationModel, 158
 - PhotonTransformViewScaleModel, 160
- DisabledPickupItems
 - PickupItem, 169
- Disconnect
 - PhotonNetwork, 112
- DisconnectByClientTimeout
 - Public API, 19
- DisconnectByServer
 - Public API, 19
- DisconnectByServerLogic
 - Public API, 19
- DisconnectByServerTimeout
 - Public API, 19
- DisconnectByServerUserLimit
 - Public API, 19
- DisconnectCause
 - Public API, 19
- Disconnected
 - Enums.cs, 224
 - Public API, 20
- Disconnecting
 - Enums.cs, 224
 - Public API, 20
- DisconnectingFromGameserver
 - Public API, 20
- DisconnectingFromMasterserver
 - Public API, 20
- DisconnectingFromNameServer
 - Public API, 20
- Discrete
 - PhotonAnimatorView, 91
- Dispose
 - PingMonoEditor, 173
- DoNotCache
 - ExitGames::Client::Photon, 29
- DocumentationLabel
 - PunWizardText, 188
- DocumentationLocation
 - PhotonEditor, 96
- DoesLayerSynchronizeTypeExist
 - PhotonAnimatorView, 91
- DoesParameterSynchronizeTypeExist
 - PhotonAnimatorView, 91
- Done
 - PhotonPingManager, 138
 - PingMonoEditor, 173
- drag
 - MoveByKeys, 71
- DragVector
 - InputToEvent, 55
- Dragging
 - InputToEvent, 55
- Draw
 - ExitGames::Client::GUI::GizmoTypeDrawer, 53
- DrawErrorGizmo
 - PhotonTransformViewPositionModel, 157
- DrawGizmoOptions

- PhotonGUI, 97
- DrawSplitter
 - PhotonGUI, 97
- Drop
 - PickupItem, 168
- EmailOrAppIdLabel
 - PunWizardText, 188
- EmptyRoomTTL
 - ExitGames::Client::Photon::ParameterCode, 85
- EnableLobbyStatistics
 - PhotonNetwork, 131
 - ServerSettings, 206
- EnabledRegions
 - ServerSettings, 206
- Encrypt
 - RaiseEventOptions, 192
- Enums.cs
 - asia, 224
 - au, 224
 - CloudRegionCode, 223
 - CloudRegionFlag, 224
 - Connected, 224
 - Connecting, 224
 - ConnectionState, 224
 - Disconnected, 224
 - Disconnecting, 224
 - eu, 224
 - GameServer, 224
 - InitializingApplication, 224
 - jp, 224
 - MasterServer, 224
 - NameServer, 224
 - none, 224
 - ServerConnection, 224
 - us, 224
- Equals
 - PhotonPlayer, 140
 - RoomInfo, 198
- ErrorInfo
 - ExitGames::Client::Photon::EventCode, 45
- ErrorTextTitle
 - PunWizardText, 188
- ErrorsOnly
 - Public API, 20
- EstimateSpeedAndTurn
 - PhotonTransformViewPositionModel, 156
- EstimatedSpeed
 - PhotonTransformViewPositionModel, 156
- eu
 - Enums.cs, 224
- EventCaching
 - ExitGames::Client::Photon, 29
- EventCallback
 - PhotonNetwork, 112
- EventForward
 - ExitGames::Client::Photon::ParameterCode, 85
- Exception
 - Public API, 19
- ExceptionOnConnect
 - Public API, 19
- ExchangeKeysForEncryption
 - ExitGames::Client::Photon::OperationCode, 78
- ExitGames, 27
- ExitGames.Client, 27
- ExitGames.Client.DemoParticle, 27
- ExitGames.Client.DemoParticle.TimeKeeper, 213
- ExitGames.Client.GUI.GizmoTypeDrawer, 53
- ExitGames.Client.GUI, 27
- ExitGames.Client.Photon, 28
- ExitGames.Client.Photon.ActorProperties, 35
- ExitGames.Client.Photon.ErrorCode, 40
- ExitGames.Client.Photon.EventCode, 44
- ExitGames.Client.Photon.GamePropertyKey, 51
- ExitGames.Client.Photon.LoadbalancingPeer.Enter↔
 - RoomParams, 39
- ExitGames.Client.Photon.LoadbalancingPeer.OpJoin↔
 - RandomRoomParams, 79
- ExitGames.Client.Photon.OperationCode, 76
- ExitGames.Client.Photon.ParameterCode, 80
- ExitGames::Client::DemoParticle::TimeKeeper
 - Interval, 214
 - IsEnabled, 214
 - Reset, 214
 - ShouldExecute, 215
 - TimeKeeper, 214
- ExitGames::Client::GUI::GizmoTypeDrawer
 - Draw, 53
- ExitGames::Client::GUI
 - Cube, 28
 - GizmoType, 28
 - Sphere, 28
 - WireCube, 28
 - WireSphere, 28
- ExitGames::Client::Photon
 - Actor, 30
 - AddToRoomCache, 29
 - AddToRoomCacheGlobal, 29
 - All, 30
 - CreateIfNotExists, 29
 - Default, 29
 - DoNotCache, 29
 - EventCaching, 29
 - FillRoom, 30
 - Game, 30
 - GameAndActor, 30
 - JoinMode, 29
 - JoinOrRejoin, 29
 - MasterClient, 30
 - MatchmakingMode, 29
 - MergeCache, 29
 - None, 30
 - Others, 30
 - PropertyTypeFlag, 30
 - RandomMatching, 30
 - ReceiverGroup, 30
 - RejoinOnly, 29

- RemoveCache, 29
- RemoveFromRoomCache, 29
- RemoveFromRoomCacheForActorsLeft, 29
- ReplaceCache, 29
- SerialMatching, 30
- SliceIncreaseIndex, 29
- SlicePurgeIndex, 29
- SlicePurgeUpToIndex, 29
- SliceSetIndex, 29
- ExitGames::Client::Photon::ActorProperties
 - IsInactive, 35
 - PlayerName, 35
 - UserId, 35
- ExitGames::Client::Photon::ErrorCode
 - AlreadyMatched, 41
 - AuthenticationTicketExpired, 41
 - CustomAuthenticationFailed, 41
 - GameClosed, 41
 - GameDoesNotExist, 41
 - GameFull, 42
 - GameIdAlreadyExists, 42
 - InternalServerError, 42
 - InvalidAuthentication, 42
 - InvalidOperation, 42
 - InvalidOperationCode, 42
 - InvalidRegion, 42
 - MaxCcuReached, 42
 - NoRandomMatchFound, 43
 - Ok, 43
 - OperationNotAllowedInCurrentState, 43
 - PluginMismatch, 43
 - PluginReportedError, 43
 - ServerFull, 43
 - UserBlocked, 43
- ExitGames::Client::Photon::EventCode
 - AppStats, 45
 - AzureNodeInfo, 45
 - CacheSliceChanged, 45
 - ErrorInfo, 45
 - GameList, 45
 - GameListUpdate, 45
 - Join, 45
 - Leave, 45
 - LobbyStats, 46
 - Match, 46
 - PropertiesChanged, 46
 - QueueState, 46
 - SetProperties, 46
- ExitGames::Client::Photon::GamePropertyKey
 - CleanupCacheOnLeave, 52
 - IsOpen, 52
 - IsVisible, 52
 - MasterClientId, 52
 - MaxPlayers, 52
 - PlayerCount, 52
 - PropsListedInLobby, 52
 - Removed, 52
- ExitGames::Client::Photon::LoadbalancingPeer::↔
 - EnterRoomParams
 - CreateIfNotExists, 40
 - Lobby, 40
 - OnGameServer, 40
 - PlayerProperties, 40
 - RoomName, 40
 - RoomOptions, 40
- ExitGames::Client::Photon::LoadbalancingPeer::Op↔
 - JoinRandomRoomParams
 - ExpectedCustomRoomProperties, 80
 - ExpectedMaxPlayers, 80
 - MatchingType, 80
 - SqlLobbyFilter, 80
 - TypedLobby, 80
- ExitGames::Client::Photon::OperationCode
 - Authenticate, 77
 - ChangeGroups, 77
 - CreateGame, 78
 - ExchangeKeysForEncryption, 78
 - FindFriends, 78
 - GetLobbyStats, 78
 - GetProperties, 78
 - GetRegions, 78
 - Join, 78
 - JoinGame, 78
 - JoinLobby, 78
 - JoinRandomGame, 78
 - Leave, 78
 - LeaveLobby, 79
 - RaiseEvent, 79
 - SetProperties, 79
 - WebRpc, 79
- ExitGames::Client::Photon::ParameterCode
 - ActorList, 83
 - ActorNr, 83
 - Add, 83
 - Address, 83
 - AppVersion, 83
 - ApplicationId, 83
 - AzureLocalNodeId, 83
 - AzureMasterNodeId, 84
 - AzureNodeInfo, 84
 - Broadcast, 84
 - Cache, 84
 - CacheSliceIndex, 84
 - CheckUserOnJoin, 84
 - CleanupCacheOnLeave, 84
 - ClientAuthenticationData, 84
 - ClientAuthenticationParams, 84
 - ClientAuthenticationType, 84
 - Code, 85
 - CustomEventContent, 85
 - Data, 85
 - EmptyRoomTTL, 85
 - EventForward, 85
 - ExpectedValues, 85
 - FindFriendsRequestList, 85

- FindFriendsResponseOnlineList, [85](#)
- FindFriendsResponseRoomIdList, [85](#)
- GameCount, [86](#)
- GameList, [86](#)
- GameProperties, [86](#)
- Group, [86](#)
- Info, [86](#)
- IsComingBack, [86](#)
- IsInactive, [86](#)
- JoinMode, [86](#)
- LobbyName, [86](#)
- LobbyStats, [87](#)
- LobbyType, [87](#)
- MasterClientId, [87](#)
- MasterPeerCount, [87](#)
- MatchMakingType, [87](#)
- PeerCount, [87](#)
- PlayerProperties, [87](#)
- PlayerTTL, [87](#)
- PluginName, [87](#)
- PluginVersion, [88](#)
- Plugins, [88](#)
- Position, [88](#)
- Properties, [88](#)
- PublishUserId, [88](#)
- ReceiverGroup, [88](#)
- Region, [88](#)
- Remove, [88](#)
- RoomName, [88](#)
- Secret, [89](#)
- SuppressRoomEvents, [89](#)
- TargetActorNr, [89](#)
- UriPath, [89](#)
- UserId, [89](#)
- WebRpcParameters, [89](#)
- WebRpcReturnCode, [89](#)
- WebRpcReturnMessage, [89](#)
- ExpectedCustomRoomProperties
 - ExitGames::Client::Photon::LoadbalancingPeer::↔
OpJoinRandomRoomParams, [80](#)
- ExpectedMaxPlayers
 - ExitGames::Client::Photon::LoadbalancingPeer::↔
OpJoinRandomRoomParams, [80](#)
- ExpectedValues
 - ExitGames::Client::Photon::ParameterCode, [85](#)
- Extensions, [46](#)
 - AlmostEquals, [47](#)
 - Contains, [48](#)
 - GetPhotonView, [48](#)
 - GetPhotonViewsInChildren, [48](#)
 - Merge, [48](#)
 - MergeStringKeys, [48](#)
 - StripKeysWithNullValues, [48](#)
 - StripToStringKeys, [49](#)
 - ToStringFull, [49](#)
- Extensions.cs
 - Hashtable, [225](#)
 - SupportClass, [225](#)
- ExtrapolateIncludingRoundTripTime
 - PhotonTransformViewPositionModel, [157](#)
- ExtrapolateNumberOfStoredPositions
 - PhotonTransformViewPositionModel, [157](#)
- ExtrapolateOption
 - PhotonTransformViewPositionModel, [157](#)
- ExtrapolateOptions
 - PhotonTransformViewPositionModel, [156](#)
- ExtrapolateSpeed
 - PhotonTransformViewPositionModel, [157](#)
- Facebook
 - LoadbalancingPeer.cs, [227](#)
- FetchServerTimestamp
 - PhotonNetwork, [113](#)
- FillRoom
 - ExitGames::Client::Photon, [30](#)
- Find
 - PhotonPlayer, [140](#)
 - PhotonView, [162](#)
- FindFriends
 - ExitGames::Client::Photon::OperationCode, [78](#)
 - PhotonNetwork, [113](#)
- FindFriendsRequestList
 - ExitGames::Client::Photon::ParameterCode, [85](#)
- FindFriendsResponseOnlineList
 - ExitGames::Client::Photon::ParameterCode, [85](#)
- FindFriendsResponseRoomIdList
 - ExitGames::Client::Photon::ParameterCode, [85](#)
- FindGameObjectsWithComponent
 - PhotonNetwork, [113](#)
- Fixed
 - PhotonView.cs, [233](#)
- FixedSpeed
 - PhotonTransformViewPositionModel, [156](#)
- FixedUpdate
 - MoveByKeys, [71](#)
- Float
 - PhotonAnimatorView, [91](#)
- FoldoutBold
 - PhotonGUI, [98](#)
- font
 - ShowInfoOfPlayer, [208](#)
- ForwardToWebhook
 - RaiseEventOptions, [192](#)
- FriendInfo, [49](#)
 - IsInRoom, [50](#)
 - IsOnline, [50](#)
 - Name, [50](#)
 - Room, [50](#)
 - ToString, [50](#)
- Friends
 - PhotonNetwork, [131](#)
- FriendsListAge
 - PhotonNetwork, [132](#)
- Full
 - Public API, [20](#)
- FullRPCListLabel
 - PunWizardText, [189](#)

- FullRPCListTitle
 - PunWizardText, [189](#)
- Game
 - ExitGames::Client::Photon, [30](#)
- GameAndActor
 - ExitGames::Client::Photon, [30](#)
- GameClosed
 - ExitGames::Client::Photon::ErrorCode, [41](#)
- GameCount
 - ExitGames::Client::Photon::ParameterCode, [86](#)
- GameDoesNotExist
 - ExitGames::Client::Photon::ErrorCode, [41](#)
- GameFull
 - ExitGames::Client::Photon::ErrorCode, [42](#)
- GameIdAlreadyExists
 - ExitGames::Client::Photon::ErrorCode, [42](#)
- GameList
 - ExitGames::Client::Photon::EventCode, [45](#)
 - ExitGames::Client::Photon::ParameterCode, [86](#)
- GameListUpdate
 - ExitGames::Client::Photon::EventCode, [45](#)
- GameObjectExtensions, [50](#)
 - GetActive, [51](#)
- GameProperties
 - ExitGames::Client::Photon::ParameterCode, [86](#)
- GameServer
 - Enums.cs, [224](#)
- gameVersion
 - PhotonNetwork, [132](#)
- Get
 - PhotonPlayer, [140](#)
 - PhotonView, [162](#)
- GetActive
 - GameObjectExtensions, [51](#)
- GetAllSubTypesInScripts
 - PhotonEditor, [95](#)
- GetExtrapolatedPositionOffset
 - PhotonTransformViewPositionControl, [154](#)
- GetHashCode
 - PhotonPlayer, [140](#)
 - RoomInfo, [198](#)
- GetID
 - PhotonViewHandler, [167](#)
- GetLayerSynchronizeType
 - PhotonAnimatorView, [91](#)
- GetLobbyStats
 - ExitGames::Client::Photon::OperationCode, [78](#)
- GetNetworkPosition
 - PhotonTransformViewPositionControl, [154](#)
- GetNext
 - PhotonPlayer, [141](#)
- GetNextFor
 - PhotonPlayer, [141](#)
- GetParameterSynchronizeType
 - PhotonAnimatorView, [92](#)
- GetPhotonView
 - Extensions, [48](#)
- GetPhotonViewsInChildren
 - Extensions, [48](#)
- GetPing
 - PhotonNetwork, [114](#)
- GetProperties
 - ExitGames::Client::Photon::OperationCode, [78](#)
- GetRegions
 - ExitGames::Client::Photon::OperationCode, [78](#)
- GetRoomList
 - PhotonNetwork, [114](#)
- GetRotation
 - PhotonTransformViewRotationControl, [158](#)
- GetScale
 - PhotonTransformViewScaleControl, [159](#)
- GetScore
 - ScoreExtensions, [204](#)
- GetScriptsInFolder
 - PhotonConverter, [94](#)
- GetSynchronizedLayers
 - PhotonAnimatorView, [92](#)
- GetSynchronizedParameters
 - PhotonAnimatorView, [92](#)
- GetTeam
 - TeamExtensions, [213](#)
- GizmoType
 - ExitGames::Client::GUI, [28](#)
- goPointedAt
 - InputToEvent, [55](#)
- Group
 - ExitGames::Client::Photon::ParameterCode, [86](#)
- group
 - PhotonView, [165](#)
- GuiRect
 - InRoomChat, [56](#)
- HasQueuedObjects
 - PhotonStreamQueue, [151](#)
- Hashtable
 - Extensions.cs, [225](#)
 - LoadbalancingPeer.cs, [227](#)
 - NetworkingPeer.cs, [228](#)
 - Photon, [31](#)
 - PhotonClasses.cs, [230](#)
 - PhotonHandler.cs, [230](#)
 - PhotonNetwork.cs, [231](#)
 - PhotonPlayer.cs, [231](#)
 - PickupItem.cs, [240](#)
 - PickupItemSyncer.cs, [240](#)
 - PunPlayerScores.cs, [241](#)
 - PunTeams.cs, [241](#)
- healthStatsVisible
 - PhotonStatsGui, [146](#)
- HelpIcon
 - PhotonGUI, [98](#)
- HelpURL, [53](#)
 - HelpURL, [54](#)
- HighlightOwnedGameObj, [54](#)
 - Offset, [54](#)
 - PointerPrefab, [54](#)
- HostAndPort

- Region, [193](#)
- HostType
 - ServerSettings, [206](#)
- HostingOption
 - ServerSettings, [205](#)
- IPunCallbacks, [58](#)
 - OnConnectedToMaster, [60](#)
 - OnConnectedToPhoton, [60](#)
 - OnConnectionFail, [60](#)
 - OnCreatedRoom, [60](#)
 - OnCustomAuthenticationFailed, [60](#)
 - OnDisconnectedFromPhoton, [62](#)
 - OnFailedToConnectToPhoton, [62](#)
 - OnJoinedLobby, [62](#)
 - OnJoinedRoom, [62](#)
 - OnLeftLobby, [62](#)
 - OnLeftRoom, [63](#)
 - OnLobbyStatisticsUpdate, [63](#)
 - OnMasterClientSwitched, [63](#)
 - OnOwnershipRequest, [63](#)
 - OnPhotonCreateRoomFailed, [64](#)
 - OnPhotonCustomRoomPropertiesChanged, [64](#)
 - OnPhotonInstantiate, [64](#)
 - OnPhotonJoinRoomFailed, [64](#)
 - OnPhotonMaxCccuReached, [65](#)
 - OnPhotonPlayerConnected, [65](#)
 - OnPhotonPlayerDisconnected, [65](#)
 - OnPhotonPlayerPropertiesChanged, [65](#)
 - OnPhotonRandomJoinFailed, [66](#)
 - OnReceivedRoomListUpdate, [66](#)
 - OnUpdatedFriendList, [66](#)
 - OnWebRpcResponse, [66](#)
- IPunObservable, [67](#)
- IPunPrefabPool, [68](#)
 - Destroy, [68](#)
 - Instantiate, [68](#)
- ID
 - PhotonPlayer, [142](#)
- IgnoreInitialAttempt
 - PhotonPingManager, [138](#)
- inRoom
 - PhotonNetwork, [132](#)
- InRoomChat, [55](#)
 - AddLine, [56](#)
 - AlignBottom, [56](#)
 - Chat, [56](#)
 - ChatRPC, [56](#)
 - GuiRect, [56](#)
 - IsVisible, [56](#)
 - messages, [56](#)
 - OnGUI, [56](#)
 - Start, [56](#)
- InRoomRoundTimer, [56](#)
 - OnGUI, [57](#)
 - OnJoinedRoom, [57](#)
 - OnMasterClientSwitched, [57](#)
 - OnPhotonCustomRoomPropertiesChanged, [57](#)
 - SecondsPerTurn, [58](#)
 - secondsbeforeend, [58](#)
 - StartTime, [58](#)
 - starting_severtime, [58](#)
 - TextPos, [58](#)
 - timetostart, [58](#)
- IncorrectRPCListLabel
 - PunWizardText, [189](#)
- IncorrectRPCListTitle
 - PunWizardText, [189](#)
- Info
 - ExitGames::Client::Photon::ParameterCode, [86](#)
- Informational
 - Public API, [20](#)
- InitializeSecurity
 - PhotonNetwork, [114](#)
- InitializingApplication
 - Enums.cs, [224](#)
- inputHitPos
 - InputToEvent, [55](#)
- InputToEvent, [54](#)
 - DetectPointedAtGameObject, [55](#)
 - DragVector, [55](#)
 - Dragging, [55](#)
 - goPointedAt, [55](#)
 - inputHitPos, [55](#)
- insideLobby
 - PhotonNetwork, [132](#)
- Instance
 - PunSceneSettings, [184](#)
- Instantiate
 - IPunPrefabPool, [68](#)
 - PhotonNetwork, [114](#), [115](#)
- InstantiateInRoomOnly
 - PhotonNetwork, [127](#)
- InstantiateRpc
 - ManualPhotonViewAllocator, [69](#)
- InstantiateSceneObject
 - PhotonNetwork, [115](#)
- InstantiateType
 - OnClickInstantiate, [74](#)
- instantiationData
 - PhotonView, [165](#)
- instantiationId
 - PhotonView, [165](#)
- Int
 - PhotonAnimatorView, [91](#)
- InterestGroup
 - RaiseEventOptions, [192](#)
- InternalReceiveException
 - Public API, [19](#)
- InternalServerError
 - ExitGames::Client::Photon::ErrorCode, [42](#)
- InterpolateLerpSpeed
 - PhotonTransformViewPositionModel, [157](#)
 - PhotonTransformViewRotationModel, [159](#)
 - PhotonTransformViewScaleModel, [160](#)
- InterpolateMoveTowardsAcceleration
 - PhotonTransformViewPositionModel, [157](#)

- InterpolateMoveTowardsDeceleration
 - PhotonTransformViewPositionModel, [157](#)
- InterpolateMoveTowardsSpeed
 - PhotonTransformViewPositionModel, [157](#)
 - PhotonTransformViewScaleModel, [160](#)
- InterpolateOption
 - PhotonTransformViewPositionModel, [157](#)
 - PhotonTransformViewRotationModel, [159](#)
 - PhotonTransformViewScaleModel, [160](#)
- InterpolateOptions
 - PhotonTransformViewPositionModel, [156](#)
 - PhotonTransformViewRotationModel, [158](#)
 - PhotonTransformViewScaleModel, [160](#)
- InterpolateRotateTowardsSpeed
 - PhotonTransformViewRotationModel, [159](#)
- InterpolateSpeedCurve
 - PhotonTransformViewPositionModel, [157](#)
- Interval
 - ExitGames::Client::DemoParticle::TimeKeeper, [214](#)
- InvalidAuthentication
 - ExitGames::Client::Photon::ErrorCode, [42](#)
 - Public API, [19](#)
- InvalidOperation
 - ExitGames::Client::Photon::ErrorCode, [42](#)
- InvalidOperationCode
 - ExitGames::Client::Photon::ErrorCode, [42](#)
- InvalidRegion
 - ExitGames::Client::Photon::ErrorCode, [42](#)
 - Public API, [19](#)
- IsAppld
 - ServerSettingsInspector, [207](#)
- IsComingBack
 - ExitGames::Client::Photon::ParameterCode, [86](#)
- IsDefault
 - TypedLobby, [216](#)
- IsEnabled
 - ExitGames::Client::DemoParticle::TimeKeeper, [214](#)
- IsGrounded
 - MoveByKeys, [71](#)
- IsInRoom
 - FriendInfo, [50](#)
- IsInactive
 - ExitGames::Client::Photon::ActorProperties, [35](#)
 - ExitGames::Client::Photon::ParameterCode, [86](#)
- isLocal
 - PhotonPlayer, [142](#)
- isLocalClientInside
 - RoomInfo, [199](#)
- isMasterClient
 - PhotonNetwork, [132](#)
 - PhotonPlayer, [142](#)
- isMessageQueueRunning
 - PhotonNetwork, [132](#)
- isMine
 - PhotonView, [165](#)
- isNonMasterClientInRoom
 - PhotonNetwork, [133](#)
- IsOnline
 - FriendInfo, [50](#)
- IsOpen
 - ExitGames::Client::Photon::GamePropertyKey, [52](#)
- isOpen
 - RoomOptions, [202](#)
- isOwnerActive
 - PhotonView, [166](#)
- isReading
 - PhotonStream, [149](#)
- isSceneView
 - PhotonView, [166](#)
- IsVisible
 - ExitGames::Client::Photon::GamePropertyKey, [52](#)
 - InRoomChat, [56](#)
- isVisible
 - RoomOptions, [202](#)
- IsWaitingForPickupInit
 - PickupItemSyncer, [172](#)
- isWriting
 - PhotonStream, [150](#)
- Join
 - ExitGames::Client::Photon::EventCode, [45](#)
 - ExitGames::Client::Photon::OperationCode, [78](#)
- JoinGame
 - ExitGames::Client::Photon::OperationCode, [78](#)
- JoinLobby
 - ExitGames::Client::Photon::OperationCode, [78](#)
 - PhotonNetwork, [116](#)
 - ServerSettings, [206](#)
- JoinMode
 - ExitGames::Client::Photon, [29](#)
 - ExitGames::Client::Photon::ParameterCode, [86](#)
- JoinOrCreateRoom
 - PhotonNetwork, [117](#)
- JoinOrRejoin
 - ExitGames::Client::Photon, [29](#)
- JoinRandomGame
 - ExitGames::Client::Photon::OperationCode, [78](#)
- JoinRandomRoom
 - PhotonNetwork, [117](#), [118](#)
- JoinRoom
 - PhotonNetwork, [118](#)
- Joined
 - Public API, [20](#)
- JoinedLobby
 - Public API, [20](#)
- Joining
 - Public API, [20](#)
- jp
 - Enums.cs, [224](#)
- JumpForce
 - MoveByKeys, [72](#)
- JumpTimeout
 - MoveByKeys, [72](#)
- lastPowerup

- MoveByKeys, 72
- LayerIndex
 - PhotonAnimatorView::SynchronizedLayer, 212
- Leave
 - ExitGames::Client::Photon::EventCode, 45
 - ExitGames::Client::Photon::OperationCode, 78
- LeaveLobby
 - ExitGames::Client::Photon::OperationCode, 79
 - PhotonNetwork, 119
- LeaveRoom
 - PhotonNetwork, 119
- Leaving
 - Public API, 20
- Lerp
 - PhotonTransformViewPositionModel, 156
 - PhotonTransformViewRotationModel, 158
 - PhotonTransformViewScaleModel, 160
- LoadAllScenesToFix
 - PhotonViewHandler, 167
- LoadLevel
 - PhotonNetwork, 119
- LoadScene
 - UnityEngine::SceneManagement::SceneManager, 203
- LoadbalancingPeer.cs
 - AsyncRandomLobby, 228
 - Custom, 227
 - CustomAuthenticationType, 227
 - Default, 228
 - Facebook, 227
 - Hashtable, 227
 - LobbyType, 227
 - None, 227
 - SqlLobby, 228
 - Steam, 227
- Lobby
 - ExitGames::Client::Photon::LoadbalancingPeer::↔
 - EnterRoomParams, 40
- lobby
 - PhotonNetwork, 133
- LobbyName
 - ExitGames::Client::Photon::ParameterCode, 86
- LobbyStatistics
 - PhotonNetwork, 133
- LobbyStats
 - ExitGames::Client::Photon::EventCode, 46
 - ExitGames::Client::Photon::ParameterCode, 87
- LobbyType
 - ExitGames::Client::Photon::ParameterCode, 87
 - LoadbalancingPeer.cs, 227
- LocateSettingsButton
 - PunWizardText, 189
- logLevel
 - PhotonNetwork, 127
- LogStats
 - SupportLogging, 211
- LogTrafficStats
 - SupportLogger, 210
- SupportLogging, 211
- MAX_VIEW_IDS
 - PhotonNetwork, 127
- MainMenuButton
 - PunWizardText, 189
- ManualPhotonViewAllocator, 69
 - AllocateManualPhotonView, 69
 - InstantiateRpc, 69
 - Prefab, 69
- MasterClient
 - ExitGames::Client::Photon, 30
 - Public API, 24
- masterClient
 - PhotonNetwork, 133
- MasterClientId
 - ExitGames::Client::Photon::GamePropertyKey, 52
 - ExitGames::Client::Photon::ParameterCode, 87
- MasterPeerCount
 - ExitGames::Client::Photon::ParameterCode, 87
- MasterServer
 - Enums.cs, 224
- Match
 - ExitGames::Client::Photon::EventCode, 46
- MatchMakingType
 - ExitGames::Client::Photon::ParameterCode, 87
- MatchingType
 - ExitGames::Client::Photon::LoadbalancingPeer::↔
 - OpJoinRandomRoomParams, 80
- MatchmakingMode
 - ExitGames::Client::Photon, 29
- MaxCcuReached
 - ExitGames::Client::Photon::ErrorCode, 42
 - Public API, 19
- maxConnections
 - PhotonNetwork, 127
- MaxMillisecondsPerPing
 - PhotonPingManager, 138
- MaxPlayers
 - ExitGames::Client::Photon::GamePropertyKey, 52
- maxPlayers
 - Room, 196
 - RoomInfo, 200
 - RoomOptions, 201
- maxPlayersField
 - RoomInfo, 199
- maxPowerups
 - MoveByKeys, 72
- MaxResendsBeforeDisconnect
 - PhotonNetwork, 133
- MenuItemHighlightSettings
 - PhotonEditor, 95
- MenuItemOpenWizard
 - PhotonEditor, 95
- Merge
 - Extensions, 48
- MergeCache
 - ExitGames::Client::Photon, 29
- MergeStringKeys

- Extensions, [48](#)
- Message
 - AccountService, [34](#)
- messages
 - InRoomChat, [56](#)
- minViewId
 - SceneSetting, [204](#)
- MinViewIdForScene
 - PunSceneSettings, [184](#)
- MinViewIdPerScene
 - PunSceneSettings, [184](#)
- MobileExportNoteLabel
 - PunWizardText, [189](#)
- MobilePunPlusExportNoteLabel
 - PunWizardText, [189](#)
- MoveByKeys, [70](#)
 - dir, [71](#)
 - dirFix, [71](#)
 - disableTime, [71](#)
 - drag, [71](#)
 - FixedUpdate, [71](#)
 - IsGrounded, [71](#)
 - JumpForce, [72](#)
 - JumpTimeout, [72](#)
 - lastPowerup, [72](#)
 - maxPowerups, [72](#)
 - powerText, [72](#)
 - powerups, [72](#)
 - rad, [72](#)
 - Start, [71](#)
 - thrust, [72](#)
- MoveTowards
 - PhotonTransformViewScaleModel, [160](#)
- Name
 - FriendInfo, [50](#)
 - PhotonAnimatorView::SynchronizedParameter, [212](#)
 - TypedLobby, [216](#)
 - WebRpcResponse, [218](#)
- name
 - PhotonPlayer, [143](#)
 - Room, [196](#)
 - RoomInfo, [200](#)
- nameField
 - RoomInfo, [199](#)
- NameServer
 - Enums.cs, [224](#)
- NetworkStatisticsEnabled
 - PhotonNetwork, [134](#)
- NetworkStatisticsReset
 - PhotonNetwork, [120](#)
- NetworkStatisticsToString
 - PhotonNetwork, [120](#)
- networkView
 - Photon::MonoBehaviour, [70](#)
- NetworkingPeer.cs
 - Hashtable, [228](#)
- NoRandomMatchFound
 - ExitGames::Client::Photon::ErrorCode, [43](#)
- None
 - ExitGames::Client::Photon, [30](#)
 - LoadbalancingPeer.cs, [227](#)
- none
 - Enums.cs, [224](#)
 - PunTeams, [185](#)
- NotSet
 - ServerSettings, [205](#)
- observed
 - PhotonView, [165](#)
- ObservedComponents
 - PhotonView, [165](#)
- Off
 - PhotonView.cs, [233](#)
- OfflineMode
 - ServerSettings, [205](#)
- offlineMode
 - PhotonNetwork, [134](#)
- Offset
 - HighlightOwnedGameObj, [54](#)
- Ok
 - ExitGames::Client::Photon::ErrorCode, [43](#)
- OkButton
 - PunWizardText, [189](#)
- OnApplicationPause
 - SupportLogging, [211](#)
- OnApplicationQuit
 - SupportLogging, [211](#)
- OnAwakeRPC
 - OnAwakeUsePhotonView, [72](#)
- OnAwakeUsePhotonView, [72](#)
 - OnAwakeRPC, [72](#)
- OnClick
 - OnClickDestroy, [73](#)
 - OnClickLoadSomething, [75](#)
- OnClickDestroy, [73](#)
 - DestroyByRpc, [73](#)
 - DestroyRpc, [73](#)
 - OnClick, [73](#)
- OnClickInstantiate, [74](#)
 - InstantiateType, [74](#)
 - Prefab, [74](#)
 - showGui, [74](#)
- OnClickLoadSomething, [74](#)
 - OnClick, [75](#)
 - ResourceToLoad, [75](#)
 - ResourceTypeOption, [75](#)
 - ResourceTypeToLoad, [75](#)
 - Scene, [75](#)
 - Web, [75](#)
- OnConnectedToMaster
 - ConnectAndJoinRandom, [39](#)
 - IPunCallbacks, [60](#)
 - Photon::PunBehaviour, [175](#)
 - Public API, [23](#)
- OnConnectedToPhoton
 - IPunCallbacks, [60](#)

- Photon::PunBehaviour, 175
- Public API, 21
- SupportLogging, 211
- OnConnectionFail
 - IPunCallbacks, 60
 - Photon::PunBehaviour, 175
 - Public API, 22
- OnCreatedRoom
 - IPunCallbacks, 60
 - Photon::PunBehaviour, 176
 - Public API, 21
 - SupportLogging, 211
- OnCustomAuthenticationFailed
 - IPunCallbacks, 60
 - Photon::PunBehaviour, 176
 - Public API, 24
- OnDisconnectedFromPhoton
 - IPunCallbacks, 62
 - Photon::PunBehaviour, 176
 - Public API, 22
 - SupportLogging, 211
- OnEnable
 - PhotonTransformViewEditor, 154
- OnEventCall
 - PhotonNetwork, 127
- OnFailedToConnectToPhoton
 - ConnectAndJoinRandom, 39
 - IPunCallbacks, 62
 - Photon::PunBehaviour, 177
 - Public API, 22
 - SupportLogging, 211
- OnGUI
 - InRoomChat, 56
 - InRoomRoundTimer, 57
 - PhotonEditor, 95
 - PhotonLagSimulationGui, 99
 - PhotonStatsGui, 145
- OnGameServer
 - ExitGames::Client::Photon::LoadbalancingPeer::↔
EnterRoomParams, 40
- OnInspectorGUI
 - PhotonAnimatorViewEditor, 93
 - PhotonRigidbody2DViewEditor, 144
 - PhotonRigidbodyViewEditor, 144
 - PhotonTransformViewEditor, 154
 - PhotonViewInspector, 167
 - ServerSettingsInspector, 207
- OnJoinedInstantiate, 75
 - OnJoinedRoom, 76
 - PlayerCamera, 76
 - PositionOffset, 76
 - PrefabsToInstantiate, 76
 - SpawnPosition, 76
- OnJoinedLobby
 - ConnectAndJoinRandom, 39
 - IPunCallbacks, 62
 - Photon::PunBehaviour, 177
 - Public API, 21
 - SupportLogging, 211
- OnJoinedRoom
 - ConnectAndJoinRandom, 39
 - IPunCallbacks, 62
 - InRoomRoundTimer, 57
 - OnJoinedInstantiate, 76
 - Photon::PunBehaviour, 177
 - PickupItemSyncer, 172
 - Public API, 22
 - PunTeams, 186
 - SupportLogging, 211
- OnLeftLobby
 - IPunCallbacks, 62
 - Photon::PunBehaviour, 177
 - Public API, 22
- OnLeftRoom
 - IPunCallbacks, 63
 - Photon::PunBehaviour, 177
 - Public API, 21
 - SupportLogging, 211
- OnLobbyStatisticsUpdate
 - IPunCallbacks, 63
 - Photon::PunBehaviour, 178
 - Public API, 24
- OnMasterClientSwitched
 - IPunCallbacks, 63
 - InRoomRoundTimer, 57
 - Photon::PunBehaviour, 178
 - Public API, 21
- OnOwnershipRequest
 - IPunCallbacks, 63
 - Photon::PunBehaviour, 178
 - Public API, 24
- OnPhotonCreateRoomFailed
 - IPunCallbacks, 64
 - Photon::PunBehaviour, 178
 - Public API, 21
- OnPhotonCustomRoomPropertiesChanged
 - IPunCallbacks, 64
 - InRoomRoundTimer, 57
 - Photon::PunBehaviour, 180
 - Public API, 23
- OnPhotonInstantiate
 - IPunCallbacks, 64
 - Photon::PunBehaviour, 180
 - Public API, 23
- OnPhotonJoinRoomFailed
 - IPunCallbacks, 64
 - Photon::PunBehaviour, 180
 - Public API, 21
- OnPhotonMaxCccuReached
 - IPunCallbacks, 65
 - Photon::PunBehaviour, 180
 - Public API, 23
- OnPhotonPlayerConnected
 - IPunCallbacks, 65
 - Photon::PunBehaviour, 181
 - PickupItemSyncer, 172

- Public API, [22](#)
- OnPhotonPlayerDisconnected
 - IPunCallbacks, [65](#)
 - Photon::PunBehaviour, [181](#)
 - Public API, [22](#)
- OnPhotonPlayerPropertiesChanged
 - IPunCallbacks, [65](#)
 - Photon::PunBehaviour, [181](#)
 - Public API, [23](#)
 - PunTeams, [186](#)
- OnPhotonRandomJoinFailed
 - ConnectAndJoinRandom, [39](#)
 - IPunCallbacks, [66](#)
 - Photon::PunBehaviour, [182](#)
 - Public API, [22](#)
- OnPhotonSerializeView
 - PhotonTransformView, [153](#)
 - PhotonTransformViewPositionControl, [155](#)
 - PhotonTransformViewRotationControl, [158](#)
 - PhotonTransformViewScaleControl, [159](#)
 - PickupItem, [169](#)
 - Public API, [23](#), [25](#)
 - SmoothSyncMovement, [209](#)
- OnPickedUpCall
 - PickupItem, [169](#)
- OnReceivedRoomListUpdate
 - IPunCallbacks, [66](#)
 - Photon::PunBehaviour, [182](#)
 - Public API, [22](#)
- OnSerializeRigidBody
 - PhotonView.cs, [232](#)
- onSerializeRigidBodyOption
 - PhotonView, [165](#)
- OnSerializeTransform
 - PhotonView.cs, [232](#)
- onSerializeTransformOption
 - PhotonView, [165](#)
- OnStartDelete, [76](#)
- OnTriggerEnter
 - PickupItem, [169](#)
 - PickupItemSimple, [171](#)
- OnUpdatedFriendList
 - IPunCallbacks, [66](#)
 - Photon::PunBehaviour, [182](#)
 - Public API, [24](#)
- OnWebRpcResponse
 - IPunCallbacks, [66](#)
 - Photon::PunBehaviour, [182](#)
 - Public API, [24](#)
- OnlyAngularVelocity
 - PhotonView.cs, [232](#)
- OnlyPosition
 - PhotonView.cs, [232](#)
- OnlyRotation
 - PhotonView.cs, [232](#)
- OnlyScale
 - PhotonView.cs, [232](#)
- OnlyVelocity
 - PhotonView.cs, [232](#)
- PhotonView.cs, [232](#)
- open
 - Room, [196](#)
 - RoomInfo, [200](#)
- OpenCloudDashboardText
 - PunWizardText, [189](#)
- OpenCloudDashboardTooltip
 - PunWizardText, [189](#)
- OpenDevNetText
 - PunWizardText, [189](#)
- OpenDevNetTooltip
 - PunWizardText, [189](#)
- openField
 - RoomInfo, [199](#)
- OpenForumText
 - PunWizardText, [189](#)
- OpenForumTooltip
 - PunWizardText, [189](#)
- OpenPDFText
 - PunWizardText, [189](#)
- OpenPDFTooltip
 - PunWizardText, [189](#)
- OperationNotAllowedInCurrentState
 - ExitGames::Client::Photon::ErrorCode, [43](#)
- Optional Gui Elements, [26](#)
- Origin
 - AccountService, [33](#)
- otherPlayers
 - PhotonNetwork, [134](#)
- Others
 - ExitGames::Client::Photon, [30](#)
 - Public API, [24](#)
- OthersBuffered
 - Public API, [24](#)
- OverrideBestCloudServer
 - PhotonNetwork, [120](#)
- OwnHostCloudCompareLabel
 - PunWizardText, [189](#)
- owner
 - PhotonView, [166](#)
- OwnerActorNr
 - PhotonView, [166](#)
- ownerId
 - PhotonView, [165](#)
- OwnershipOption
 - PhotonView.cs, [232](#)
- ownershipTransfer
 - PhotonView, [165](#)
- PUNNameReplaceLabel
 - PunWizardText, [189](#)
- PUNNameReplaceTitle
 - PunWizardText, [189](#)
- PUNWizardLabel
 - PunWizardText, [190](#)
- PacketLossByCrcCheck
 - PhotonNetwork, [134](#)
- ParameterType
 - PhotonAnimatorView, [91](#)

- Parameters
 - WebRpcResponse, 218
- Parse
 - Region, 193
- PeekNext
 - PhotonStream, 148
- Peer
 - PhotonLagSimulationGui, 99
- PeerCount
 - ExitGames::Client::Photon::ParameterCode, 87
- PeerCreated
 - Public API, 20
- PeerState
 - Public API, 19
- Photon, 30
 - Hashtable, 31
- Photon.MonoBehaviour, 70
- Photon.PunBehaviour, 173
- Photon::MonoBehaviour
 - networkView, 70
 - photonView, 70
- Photon::PunBehaviour
 - OnConnectedToMaster, 175
 - OnConnectedToPhoton, 175
 - OnConnectionFail, 175
 - OnCreatedRoom, 176
 - OnCustomAuthenticationFailed, 176
 - OnDisconnectedFromPhoton, 176
 - OnFailedToConnectToPhoton, 177
 - OnJoinedLobby, 177
 - OnJoinedRoom, 177
 - OnLeftLobby, 177
 - OnLeftRoom, 177
 - OnLobbyStatisticsUpdate, 178
 - OnMasterClientSwitched, 178
 - OnOwnershipRequest, 178
 - OnPhotonCreateRoomFailed, 178
 - OnPhotonCustomRoomPropertiesChanged, 180
 - OnPhotonInstantiate, 180
 - OnPhotonJoinRoomFailed, 180
 - OnPhotonMaxCccuReached, 180
 - OnPhotonPlayerConnected, 181
 - OnPhotonPlayerDisconnected, 181
 - OnPhotonPlayerPropertiesChanged, 181
 - OnPhotonRandomJoinFailed, 182
 - OnReceivedRoomListUpdate, 182
 - OnUpdatedFriendList, 182
 - OnWebRpcResponse, 182
- PhotonAnimatorView, 90
 - Bool, 91
 - Continuous, 91
 - Disabled, 91
 - Discrete, 91
 - DoesLayerSynchronizeTypeExist, 91
 - DoesParameterSynchronizeTypeExist, 91
 - Float, 91
 - GetLayerSynchronizeType, 91
 - GetParameterSynchronizeType, 92
 - GetSynchronizedLayers, 92
 - GetSynchronizedParameters, 92
 - Int, 91
 - ParameterType, 91
 - SetLayerSynchronized, 92
 - SetParameterSynchronized, 93
 - SynchronizeType, 91
 - Trigger, 91
- PhotonAnimatorView.SynchronizedLayer, 211
- PhotonAnimatorView.SynchronizedParameter, 212
- PhotonAnimatorView::SynchronizedLayer
 - LayerIndex, 212
 - SynchronizeType, 212
- PhotonAnimatorView::SynchronizedParameter
 - Name, 212
 - SynchronizeType, 212
 - Type, 212
- PhotonAnimatorViewEditor, 93
 - OnInspectorGUI, 93
- PhotonClasses.cs
 - Hashtable, 230
- PhotonCloud
 - ServerSettings, 205
- PhotonConverter, 93
 - ConvertRpcAttribute, 94
 - GetScriptsInFolder, 94
 - PickFolderAndConvertScripts, 94
 - RunConversion, 94
- PhotonEditor, 94
 - ClearRpcList, 95
 - CurrentLang, 96
 - DocumentationLocation, 96
 - GetAllSubTypesInScripts, 95
 - MenuItemHighlightSettings, 95
 - MenuItemOpenWizard, 95
 - OnGUI, 95
 - PhotonEditor, 95
 - RegisterOrigin, 96
 - RegisterWithEmail, 95
 - scrollPos, 96
 - ShowRegistrationWizard, 95
 - UiMainWizard, 95
 - UiSetupApp, 96
 - Update, 96
 - UpdateRpcList, 96
 - UrlAccountPage, 96
 - UrlAppIDExplained, 96
 - UrlCloudDashboard, 96
 - UrlCompare, 96
 - UrlDevNet, 96
 - UrlForum, 96
 - UrlFreeLicense, 96
 - UrlHowToSetup, 96
 - WindowType, 96
- PhotonGUI, 97
 - AddButton, 97
 - ContainerBody, 97
 - ContainerHeader, 97

- ContainerHeaderFoldout, [97](#)
- ContainerHeaderToggle, [97](#)
- DefaultAddButtonStyle, [97](#)
- DefaultContainerRowStyle, [97](#)
- DefaultContainerStyle, [98](#)
- DefaultRemoveButtonStyle, [98](#)
- DefaultTitleStyle, [98](#)
- DrawGizmoOptions, [97](#)
- DrawSplitter, [97](#)
- FoldoutBold, [98](#)
- HelpIcon, [98](#)
- RichLabel, [98](#)
- PhotonHandler.cs
 - Debug, [230](#)
 - Hashtable, [230](#)
- PhotonLagSimulationGui, [98](#)
 - OnGUI, [99](#)
 - Peer, [99](#)
 - Start, [99](#)
 - Visible, [99](#)
 - WindowId, [99](#)
 - WindowRect, [99](#)
- PhotonLogLevel
 - Public API, [20](#)
- PhotonMessageInfo, [99](#)
 - PhotonMessageInfo, [100](#)
 - photonView, [100](#)
 - sender, [100](#)
 - timestamp, [100](#)
 - ToString, [100](#)
- PhotonNetwork, [101](#)
 - AllocateSceneViewID, [106](#)
 - AllocateViewID, [106](#)
 - AuthValues, [129](#)
 - autoCleanUpPlayerObjects, [129](#)
 - autoJoinLobby, [129](#)
 - automaticallySyncScene, [130](#)
 - BackgroundTimeout, [127](#)
 - CacheSendMonoMessageTargets, [106](#)
 - CloseConnection, [107](#)
 - ConnectToBestCloudServer, [107](#)
 - ConnectToMaster, [108](#)
 - ConnectToRegion, [108](#)
 - ConnectUsingSettings, [108](#)
 - connected, [130](#)
 - connectedAndReady, [130](#)
 - connecting, [130](#)
 - connectionState, [130](#)
 - connectionStateDetailed, [130](#)
 - countOfPlayers, [131](#)
 - countOfPlayersInRooms, [131](#)
 - countOfPlayersOnMaster, [131](#)
 - countOfRooms, [131](#)
 - CrcCheckEnabled, [131](#)
 - CreateRoom, [109](#)
 - Destroy, [110](#)
 - DestroyAll, [111](#)
 - DestroyPlayerObjects, [111, 112](#)
 - Disconnect, [112](#)
 - EnableLobbyStatistics, [131](#)
 - EventCallback, [112](#)
 - FetchServerTimestamp, [113](#)
 - FindFriends, [113](#)
 - FindGameObjectsWithComponent, [113](#)
 - Friends, [131](#)
 - FriendsListAge, [132](#)
 - gameVersion, [132](#)
 - GetPing, [114](#)
 - GetRoomList, [114](#)
 - inRoom, [132](#)
 - InitializeSecurity, [114](#)
 - insideLobby, [132](#)
 - Instantiate, [114, 115](#)
 - InstantiateInRoomOnly, [127](#)
 - InstantiateSceneObject, [115](#)
 - isMasterClient, [132](#)
 - isMessageQueueRunning, [132](#)
 - isNonMasterClientInRoom, [133](#)
 - JoinLobby, [116](#)
 - JoinOrCreateRoom, [117](#)
 - JoinRandomRoom, [117, 118](#)
 - JoinRoom, [118](#)
 - LeaveLobby, [119](#)
 - LeaveRoom, [119](#)
 - LoadLevel, [119](#)
 - lobby, [133](#)
 - LobbyStatistics, [133](#)
 - logLevel, [127](#)
 - MAX_VIEW_IDS, [127](#)
 - masterClient, [133](#)
 - maxConnections, [127](#)
 - MaxResendsBeforeDisconnect, [133](#)
 - NetworkStatisticsEnabled, [134](#)
 - NetworkStatisticsReset, [120](#)
 - NetworkStatisticsToString, [120](#)
 - offlineMode, [134](#)
 - OnEventCall, [127](#)
 - otherPlayers, [134](#)
 - OverrideBestCloudServer, [120](#)
 - PacketLossByCrcCheck, [134](#)
 - PhotonServerSettings, [127](#)
 - player, [134](#)
 - playerList, [134](#)
 - playerName, [135](#)
 - precisionForFloatSynchronization, [128](#)
 - precisionForQuaternionSynchronization, [128](#)
 - precisionForVectorSynchronization, [128](#)
 - PrefabCache, [128](#)
 - PrefabPool, [135](#)
 - QuickResends, [135](#)
 - RaiseEvent, [120](#)
 - RefreshCloudServerRating, [121](#)
 - RemovePlayerCustomProperties, [121](#)
 - RemoveRPCs, [122](#)
 - RemoveRPCsInGroup, [122](#)
 - ResentReliableCommands, [135](#)

- room, [135](#)
- SendMonoMessageTargetType, [128](#)
- SendMonoMessageTargets, [128](#)
- SendOutgoingCommands, [123](#)
- sendRate, [136](#)
- sendRateOnSerialize, [136](#)
- Server, [136](#)
- ServerAddress, [136](#)
- ServerTimestamp, [136](#)
- SetLevelPrefix, [123](#)
- SetMasterClient, [123](#)
- SetPlayerCustomProperties, [124](#)
- SetReceivingEnabled, [124](#), [125](#)
- SetSendingEnabled, [125](#)
- SwitchToProtocol, [125](#)
- time, [136](#)
- UnAllocateViewID, [126](#)
- unreliableCommandsLimit, [137](#)
- UsePrefabCache, [129](#)
- UseRpcMonoBehaviourCache, [129](#)
- versionPUN, [129](#)
- WebRpc, [126](#)
- PhotonNetwork.cs
 - Debug, [231](#)
 - Hashtable, [231](#)
- PhotonNetworkingMessage
 - Public API, [20](#)
- PhotonPingManager, [137](#)
 - Attempts, [138](#)
 - BestRegion, [138](#)
 - Done, [138](#)
 - IgnoreInitialAttempt, [138](#)
 - MaxMillisecondsPerPing, [138](#)
 - PingSocket, [138](#)
 - ResolveHost, [138](#)
 - UseNative, [138](#)
- PhotonPlayer, [139](#)
 - allProperties, [142](#)
 - customProperties, [142](#)
 - Equals, [140](#)
 - Find, [140](#)
 - Get, [140](#)
 - GetHashCode, [140](#)
 - GetNext, [141](#)
 - GetNextFor, [141](#)
 - ID, [142](#)
 - isLocal, [142](#)
 - isMasterClient, [142](#)
 - name, [143](#)
 - PhotonPlayer, [140](#)
 - SetCustomProperties, [141](#)
 - TagObject, [142](#)
 - ToString, [142](#)
 - ToStringFull, [142](#)
- PhotonPlayer.cs
 - Hashtable, [231](#)
- PhotonRigidbody2DView, [143](#)
- PhotonRigidbody2DViewEditor, [143](#)
 - OnInspectorGUI, [144](#)
- PhotonRigidbodyView, [144](#)
- PhotonRigidbodyViewEditor, [144](#)
 - OnInspectorGUI, [144](#)
- PhotonServerSettings
 - PhotonNetwork, [127](#)
- PhotonStatsGui, [145](#)
 - buttonsOn, [146](#)
 - healthStatsVisible, [146](#)
 - OnGUI, [145](#)
 - Start, [145](#)
 - statsOn, [146](#)
 - statsRect, [146](#)
 - statsWindowOn, [146](#)
 - trafficStatsOn, [146](#)
 - TrafficStatsWindow, [145](#)
 - Update, [145](#)
 - WindowId, [146](#)
- PhotonStream, [146](#)
 - Count, [149](#)
 - isReading, [149](#)
 - isWriting, [150](#)
 - PeekNext, [148](#)
 - PhotonStream, [148](#)
 - ReceiveNext, [148](#)
 - SendNext, [148](#)
 - Serialize, [148](#), [149](#)
 - ToArray, [149](#)
- PhotonStreamQueue, [150](#)
 - Deserialize, [151](#)
 - HasQueuedObjects, [151](#)
 - PhotonStreamQueue, [150](#)
 - ReceiveNext, [151](#)
 - Reset, [151](#)
 - SendNext, [151](#)
 - Serialize, [152](#)
- PhotonTargets
 - Public API, [24](#)
- PhotonTransformView, [152](#)
 - OnPhotonSerializeView, [153](#)
 - SetSynchronizedValues, [153](#)
- PhotonTransformViewEditor, [153](#)
 - OnEnable, [154](#)
 - OnInspectorGUI, [154](#)
- PhotonTransformViewPositionControl, [154](#)
 - GetExtrapolatedPositionOffset, [154](#)
 - GetNetworkPosition, [154](#)
 - OnPhotonSerializeView, [155](#)
 - PhotonTransformViewPositionControl, [154](#)
 - SetSynchronizedValues, [155](#)
 - UpdatePosition, [155](#)
- PhotonTransformViewPositionModel, [155](#)
 - Disabled, [156](#)
 - DrawErrorGizmo, [157](#)
 - EstimateSpeedAndTurn, [156](#)
 - EstimatedSpeed, [156](#)
 - ExtrapolateIncludingRoundTripTime, [157](#)
 - ExtrapolateNumberOfStoredPositions, [157](#)

- ExtrapolateOption, 157
- ExtrapolateOptions, 156
- ExtrapolateSpeed, 157
- FixedSpeed, 156
- InterpolateLerpSpeed, 157
- InterpolateMoveTowardsAcceleration, 157
- InterpolateMoveTowardsDeceleration, 157
- InterpolateMoveTowardsSpeed, 157
- InterpolateOption, 157
- InterpolateOptions, 156
- InterpolateSpeedCurve, 157
- Lerp, 156
- SynchronizeEnabled, 157
- SynchronizeValues, 156
- TeleportEnabled, 157
- TeleportIfDistanceGreaterThan, 157
- PhotonTransformViewRotationControl, 158
 - GetRotation, 158
 - OnPhotonSerializeView, 158
 - PhotonTransformViewRotationControl, 158
- PhotonTransformViewRotationModel, 158
 - Disabled, 158
 - InterpolateLerpSpeed, 159
 - InterpolateOption, 159
 - InterpolateOptions, 158
 - InterpolateRotateTowardsSpeed, 159
 - Lerp, 158
 - RotateTowards, 158
 - SynchronizeEnabled, 159
- PhotonTransformViewScaleControl, 159
 - GetScale, 159
 - OnPhotonSerializeView, 159
 - PhotonTransformViewScaleControl, 159
- PhotonTransformViewScaleModel, 159
 - Disabled, 160
 - InterpolateLerpSpeed, 160
 - InterpolateMoveTowardsSpeed, 160
 - InterpolateOption, 160
 - InterpolateOptions, 160
 - Lerp, 160
 - MoveTowards, 160
 - SynchronizeEnabled, 160
- PhotonView, 160
 - CreatorActorNr, 165
 - DeserializeView, 162
 - Find, 162
 - Get, 162
 - group, 165
 - instantiationData, 165
 - instantiationId, 165
 - isMine, 165
 - isOwnerActive, 166
 - isSceneView, 166
 - observed, 165
 - ObservedComponents, 165
 - onSerializeRigidBodyOption, 165
 - onSerializeTransformOption, 165
 - owner, 166
 - OwnerActorNr, 166
 - ownerId, 165
 - ownershipTransfer, 165
 - prefix, 166
 - prefixBackup, 165
 - RPC, 163
 - RefreshRpcMonoBehaviourCache, 162
 - RequestOwnership, 162
 - RpcSecure, 163, 164
 - SerializeView, 164
 - synchronization, 165
 - ToString, 164
 - TransferOwnership, 164
 - viewID, 166
- photonView
 - Photon::MonoBehaviour, 70
 - PhotonMessageInfo, 100
- PhotonView.cs
 - All, 232
 - Fixed, 233
 - Off, 233
 - OnSerializeRigidBody, 232
 - OnSerializeTransform, 232
 - OnlyAngularVelocity, 232
 - OnlyPosition, 232
 - OnlyRotation, 232
 - OnlyScale, 232
 - OnlyVelocity, 232
 - OwnershipOption, 232
 - PositionAndRotation, 232
 - ReliableDeltaCompressed, 233
 - Request, 233
 - Takeover, 233
 - Unreliable, 233
 - UnreliableOnChange, 233
 - ViewSynchronization, 233
- PhotonViewHandler, 166
 - GetID, 167
 - LoadAllScenesToFix, 167
- PhotonViewHandler.cs
 - Debug, 220
- PhotonViewInspector, 167
 - OnInspectorGUI, 167
- PickFolderAndConvertScripts
 - PhotonConverter, 94
- Pickup
 - PickupItem, 169
 - PickupItemSimple, 171
- PickupsMine
 - PickupItem, 169
- PickupItem, 167
 - DisabledPickupItems, 169
 - Drop, 168
 - OnPhotonSerializeView, 169
 - OnPickedUpCall, 169
 - OnTriggerEnter, 169
 - Pickup, 169
 - PickupsMine, 169

- PickupOnTrigger, 170
- PunPickup, 169
- SecondsBeforeRespawn, 170
- SentPickup, 170
- TimeOfRespawn, 170
- ViewID, 170
- PickupItem.cs
 - Hashtable, 240
- PickupItemInit
 - PickupItemSyncer, 172
- PickupItemSimple, 171
 - OnTriggerEnter, 171
 - Pickup, 171
 - PickupOnCollide, 171
 - PunPickupSimple, 171
 - RespawnAfter, 171
 - SecondsBeforeRespawn, 171
 - SentPickup, 171
- PickupItemSyncer, 172
 - AskForPickupItemSpawnTimes, 172
 - IsWaitingForPickupInit, 172
 - OnJoinedRoom, 172
 - OnPhotonPlayerConnected, 172
 - PickupItemInit, 172
 - RequestForPickupTimes, 172
- PickupItemSyncer.cs
 - Hashtable, 240
- PickupOnCollide
 - PickupItemSimple, 171
- PickupOnTrigger
 - PickupItem, 170
- Ping
 - Region, 193
- PingCloudRegions.cs
 - Debug, 233
- PingMonoEditor, 173
 - Dispose, 173
 - Done, 173
 - StartPing, 173
- PingSocket
 - PhotonPingManager, 138
- player
 - PhotonNetwork, 134
- PlayerCamera
 - OnJoinedInstantiate, 76
- PlayerCount
 - ExitGames::Client::Photon::GamePropertyKey, 52
 - TypedLobbyInfo, 217
- playerCount
 - Room, 196
 - RoomInfo, 200
- playerList
 - PhotonNetwork, 134
- PlayerName
 - ExitGames::Client::Photon::ActorProperties, 35
- playerName
 - PhotonNetwork, 135
- PlayerProperties
 - ExitGames::Client::Photon::LoadbalancingPeer::EnterRoomParams, 40
 - ExitGames::Client::Photon::ParameterCode, 87
- PlayerScoreProp
 - PunPlayerScores, 183
- PlayerTTL
 - ExitGames::Client::Photon::ParameterCode, 87
- PlayersPerTeam
 - PunTeams, 186
- Playmaker
 - AccountService, 33
- PluginMismatch
 - ExitGames::Client::Photon::ErrorCode, 43
- PluginName
 - ExitGames::Client::Photon::ParameterCode, 87
- PluginReportedError
 - ExitGames::Client::Photon::ErrorCode, 43
- PluginVersion
 - ExitGames::Client::Photon::ParameterCode, 88
- Plugins
 - ExitGames::Client::Photon::ParameterCode, 88
- PointedAtGameObjectInfo, 173
- PointerPrefab
 - HighlightOwnedGameObj, 54
- Position
 - ExitGames::Client::Photon::ParameterCode, 88
- PositionAndRotation
 - PhotonView.cs, 232
- PositionOffset
 - OnJoinedInstantiate, 76
- powerText
 - MoveByKeys, 72
- powerups
 - MoveByKeys, 72
- precisionForFloatSynchronization
 - PhotonNetwork, 128
- precisionForQuaternionSynchronization
 - PhotonNetwork, 128
- precisionForVectorSynchronization
 - PhotonNetwork, 128
- Prefab
 - ManualPhotonViewAllocator, 69
 - OnClickInstantiate, 74
- PrefabCache
 - PhotonNetwork, 128
- PrefabPool
 - PhotonNetwork, 135
- PrefabsToInstantiate
 - OnJoinedInstantiate, 76
- PreferredRegion
 - ServerSettings, 206
- prefix
 - PhotonView, 166
- prefixBackup
 - PhotonView, 165
- Properties
 - ExitGames::Client::Photon::ParameterCode, 88
- PropertiesChanged

- ExitGames::Client::Photon::EventCode, 46
- propertiesListedInLobby
 - Room, 197
- PropertyTypeFlag
 - ExitGames::Client::Photon, 30
- PropsListedInLobby
 - ExitGames::Client::Photon::GamePropertyKey, 52
- Protocol
 - ServerSettings, 206
- ProtocolChoices
 - ServerSettingsInspector, 207
- Public API, 17
 - All, 24
 - AllBuffered, 24
 - AllBufferedViaServer, 25
 - AllViaServer, 25
 - Authenticated, 20
 - Authenticating, 20
 - AuthenticationTicketExpired, 19
 - ConnectedToGameserver, 20
 - ConnectedToMaster, 20
 - ConnectedToNameServer, 20
 - ConnectingToGameserver, 20
 - ConnectingToMasterserver, 20
 - ConnectingToNameServer, 20
 - DisconnectByClientTimeout, 19
 - DisconnectByServer, 19
 - DisconnectByServerLogic, 19
 - DisconnectByServerTimeout, 19
 - DisconnectByServerUserLimit, 19
 - DisconnectCause, 19
 - Disconnected, 20
 - Disconnecting, 20
 - DisconnectingFromGameserver, 20
 - DisconnectingFromMasterserver, 20
 - DisconnectingFromNameServer, 20
 - ErrorsOnly, 20
 - Exception, 19
 - ExceptionOnConnect, 19
 - Full, 20
 - Informational, 20
 - InternalReceiveException, 19
 - InvalidAuthentication, 19
 - InvalidRegion, 19
 - Joined, 20
 - JoinedLobby, 20
 - Joining, 20
 - Leaving, 20
 - MasterClient, 24
 - MaxCcuReached, 19
 - OnConnectedToMaster, 23
 - OnConnectedToPhoton, 21
 - OnConnectionFail, 22
 - OnCreatedRoom, 21
 - OnCustomAuthenticationFailed, 24
 - OnDisconnectedFromPhoton, 22
 - OnFailedToConnectToPhoton, 22
 - OnJoinedLobby, 21
 - OnJoinedRoom, 22
 - OnLeftLobby, 22
 - OnLeftRoom, 21
 - OnLobbyStatisticsUpdate, 24
 - OnMasterClientSwitched, 21
 - OnOwnershipRequest, 24
 - OnPhotonCreateRoomFailed, 21
 - OnPhotonCustomRoomPropertiesChanged, 23
 - OnPhotonInstantiate, 23
 - OnPhotonJoinRoomFailed, 21
 - OnPhotonMaxCcuReached, 23
 - OnPhotonPlayerConnected, 22
 - OnPhotonPlayerDisconnected, 22
 - OnPhotonPlayerPropertiesChanged, 23
 - OnPhotonRandomJoinFailed, 22
 - OnPhotonSerializeView, 23, 25
 - OnReceivedRoomListUpdate, 22
 - OnUpdatedFriendList, 24
 - OnWebRpcResponse, 24
 - Others, 24
 - OthersBuffered, 24
 - PeerCreated, 20
 - PeerState, 19
 - PhotonLogLevel, 20
 - PhotonNetworkingMessage, 20
 - PhotonTargets, 24
 - Queued, 20
 - QueuedComingFromGameserver, 20
 - SecurityExceptionOnConnect, 19
 - TimeoutDisconnect, 19
 - Uninitialized, 20
- PublishUserId
 - ExitGames::Client::Photon::ParameterCode, 88
- Pun
 - AccountService, 33
- PunPickup
 - PickupItem, 169
- PunPickupSimple
 - PickupItemSimple, 171
- PunPlayerScores, 183
 - PlayerScoreProp, 183
- PunPlayerScores.cs
 - Hashtable, 241
- PunRPC, 183
- PunSceneSettings, 184
 - Instance, 184
 - MinViewIdForScene, 184
 - MinViewIdPerScene, 184
 - PunSceneSettingsCsPath, 184
- PunSceneSettingsCsPath
 - PunSceneSettings, 184
- PunTeams, 185
 - blue, 185
 - none, 185
 - OnJoinedRoom, 186
 - OnPhotonPlayerPropertiesChanged, 186
 - PlayersPerTeam, 186
 - red, 185

- Start, [186](#)
- Team, [185](#)
- TeamPlayerProp, [186](#)
- UpdateTeams, [186](#)
- PunTeams.cs
 - Hashtable, [241](#)
- PunWizardText, [187](#)
 - AlreadyRegisteredInfo, [188](#)
 - AppliedToSettingsInfo, [188](#)
 - CancelButton, [188](#)
 - CloseWindowButton, [188](#)
 - ComparisonPageButton, [188](#)
 - ConnectionInfo, [188](#)
 - ConnectionTitle, [188](#)
 - ConverterLabel, [188](#)
 - DocumentationLabel, [188](#)
 - EmailOrAppIdLabel, [188](#)
 - ErrorTextTitle, [188](#)
 - FullRPCListLabel, [189](#)
 - FullRPCListTitle, [189](#)
 - IncorrectRPCListLabel, [189](#)
 - IncorrectRPCListTitle, [189](#)
 - LocateSettingsButton, [189](#)
 - MainMenuButton, [189](#)
 - MobileExportNoteLabel, [189](#)
 - MobilePunPlusExportNoteLabel, [189](#)
 - OkButton, [189](#)
 - OpenCloudDashboardText, [189](#)
 - OpenCloudDashboardTooltip, [189](#)
 - OpenDevNetText, [189](#)
 - OpenDevNetTooltip, [189](#)
 - OpenForumText, [189](#)
 - OpenForumTooltip, [189](#)
 - OpenPDFText, [189](#)
 - OpenPDFTooltip, [189](#)
 - OwnHostCloudCompareLabel, [189](#)
 - PUNNameReplaceLabel, [189](#)
 - PUNNameReplaceTitle, [189](#)
 - PUNWizardLabel, [190](#)
 - RPCListCleared, [190](#)
 - RegisteredNewAccountInfo, [190](#)
 - RemoveOutdatedRPCsLabel, [190](#)
 - RpcFoundDialogTitle, [190](#)
 - RpcFoundMessage, [190](#)
 - RpcReplaceButton, [190](#)
 - RpcSkipReplace, [190](#)
 - ServerSettingsCleanedWarning, [190](#)
 - SettingsButton, [190](#)
 - SettingsHighlightLabel, [190](#)
 - SetupButton, [190](#)
 - SetupCompleteInfo, [190](#)
 - SetupServerCloudLabel, [190](#)
 - SetupWizardInfo, [190](#)
 - SetupWizardTitle, [190](#)
 - SetupWizardWarningMessage, [190](#)
 - SetupWizardWarningTitle, [190](#)
 - SkipButton, [190](#)
 - SkipRPCListUpdateLabel, [191](#)
 - SkipRegistrationInfo, [190](#)
 - StartButton, [191](#)
 - UNtoPUNLabel, [191](#)
 - WarningPhotonDisconnect, [191](#)
 - WindowTitle, [191](#)
 - WizardMainWindowInfo, [191](#)
- QueueState
 - ExitGames::Client::Photon::EventCode, [46](#)
- Queued
 - Public API, [20](#)
- QueuedComingFromGameserver
 - Public API, [20](#)
- QuickResends
 - PhotonNetwork, [135](#)
- QuitOnEscapeOrBack, [191](#)
- RPCListCleared
 - PunWizardText, [190](#)
- RPC
 - PhotonView, [163](#)
- rad
 - MoveByKeys, [72](#)
- RaiseEvent
 - ExitGames::Client::Photon::OperationCode, [79](#)
 - PhotonNetwork, [120](#)
- RaiseEventOptions, [191](#)
 - CachingOption, [192](#)
 - Default, [192](#)
 - Encrypt, [192](#)
 - ForwardToWebhook, [192](#)
 - InterestGroup, [192](#)
 - Receivers, [192](#)
 - SequenceChannel, [192](#)
 - TargetActors, [192](#)
- RandomMatching
 - ExitGames::Client::Photon, [30](#)
- ReceiveNext
 - PhotonStream, [148](#)
 - PhotonStreamQueue, [151](#)
- ReceiverGroup
 - ExitGames::Client::Photon, [30](#)
 - ExitGames::Client::Photon::ParameterCode, [88](#)
- Receivers
 - RaiseEventOptions, [192](#)
- red
 - PunTeams, [185](#)
- RefreshCloudServerRating
 - PhotonNetwork, [121](#)
- RefreshRpcMonoBehaviourCache
 - PhotonView, [162](#)
- Region, [193](#)
 - Code, [193](#)
 - ExitGames::Client::Photon::ParameterCode, [88](#)
 - HostAndPort, [193](#)
 - Parse, [193](#)
 - Ping, [193](#)
 - ToString, [193](#)
- RegisterByEmail

- AccountService, 34
- RegisterByEmailAsync
 - AccountService, 34
- RegisterOrigin
 - PhotonEditor, 96
- RegisterWithEmail
 - PhotonEditor, 95
- RegisteredNewAccountInfo
 - PunWizardText, 190
- RejoinOnly
 - ExitGames::Client::Photon, 29
- ReliableDeltaCompressed
 - PhotonView.cs, 233
- Remove
 - ExitGames::Client::Photon::ParameterCode, 88
- RemoveCache
 - ExitGames::Client::Photon, 29
- RemoveFromRoomCache
 - ExitGames::Client::Photon, 29
- RemoveFromRoomCacheForActorsLeft
 - ExitGames::Client::Photon, 29
- RemoveOutdatedRPCsLabel
 - PunWizardText, 190
- RemovePlayerCustomProperties
 - PhotonNetwork, 121
- RemoveRPCs
 - PhotonNetwork, 122
- RemoveRPCsInGroup
 - PhotonNetwork, 122
- Removed
 - ExitGames::Client::Photon::GamePropertyKey, 52
- removedFromList
 - RoomInfo, 200
- ReplaceCache
 - ExitGames::Client::Photon, 29
- Request
 - PhotonView.cs, 233
- RequestForPickupTimes
 - PickupItemSyncer, 172
- RequestOwnership
 - PhotonView, 162
- ResentReliableCommands
 - PhotonNetwork, 135
- Reset
 - ExitGames::Client::DemoParticle::TimeKeeper, 214
 - PhotonStreamQueue, 151
- ResolveHost
 - PhotonPingManager, 138
- ResourceToLoad
 - OnClickLoadSomething, 75
- ResourceTypeOption
 - OnClickLoadSomething, 75
- ResourceTypeToLoad
 - OnClickLoadSomething, 75
- RespawnAfter
 - PickupItemSimple, 171
- ReturnCode
 - AccountService, 34
 - WebRpcResponse, 218
- RichLabel
 - PhotonGUI, 98
- Room, 194
 - autoCleanUp, 196
 - FriendInfo, 50
 - maxPlayers, 196
 - name, 196
 - open, 196
 - playerCount, 196
 - propertiesListedInLobby, 197
 - SetCustomProperties, 195
 - SetPropertiesListedInLobby, 195
 - ToString, 196
 - ToStringFull, 196
 - visible, 197
- room
 - PhotonNetwork, 135
- RoomCount
 - TypedLobbyInfo, 217
- RoomInfo, 197
 - autoCleanUpField, 199
 - customProperties, 199
 - Equals, 198
 - GetHashCode, 198
 - isLocalClientInside, 199
 - maxPlayers, 200
 - maxPlayersField, 199
 - name, 200
 - nameField, 199
 - open, 200
 - openField, 199
 - playerCount, 200
 - removedFromList, 200
 - ToString, 198
 - ToStringFull, 198
 - visible, 200
 - visibleField, 199
- RoomName
 - ExitGames::Client::Photon::LoadbalancingPeer::↔
EnterRoomParams, 40
 - ExitGames::Client::Photon::ParameterCode, 88
- RoomOptions, 201
 - cleanupCacheOnLeave, 202
 - customRoomProperties, 201
 - customRoomPropertiesForLobby, 201
 - ExitGames::Client::Photon::LoadbalancingPeer::↔
EnterRoomParams, 40
 - isOpen, 202
 - isVisible, 202
 - maxPlayers, 201
 - suppressRoomEvents, 202
- RotateTowards
 - PhotonTransformViewRotationModel, 158
- Rotorz, 31
- Rotorz.ReorderableList, 31
- Rotorz.ReorderableList.Internal, 31

- RpcFoundDialogTitle
 - PunWizardText, [190](#)
- RpcFoundMessage
 - PunWizardText, [190](#)
- RpcList
 - ServerSettings, [206](#)
- RpcReplaceButton
 - PunWizardText, [190](#)
- RpcSecure
 - PhotonView, [163](#), [164](#)
- RpcSkipReplace
 - PunWizardText, [190](#)
- RunConversion
 - PhotonConverter, [94](#)
- Scene
 - OnClickLoadSomething, [75](#)
- SceneManagerHelper, [203](#)
 - ActiveSceneBuildIndex, [203](#)
 - ActiveSceneName, [203](#)
- sceneName
 - SceneSetting, [204](#)
- SceneSetting, [204](#)
 - minViewId, [204](#)
 - sceneName, [204](#)
- ScoreExtensions, [204](#)
 - AddScore, [204](#)
 - GetScore, [204](#)
 - SetScore, [204](#)
- scrollPos
 - PhotonEditor, [96](#)
- SecondsBeforeRespawn
 - PickupItem, [170](#)
 - PickupItemSimple, [171](#)
- SecondsPerTurn
 - InRoomRoundTimer, [58](#)
- secondsbeforeend
 - InRoomRoundTimer, [58](#)
- Secret
 - ExitGames::Client::Photon::ParameterCode, [89](#)
- SecurityExceptionOnConnect
 - Public API, [19](#)
- SelfHosted
 - ServerSettings, [205](#)
- SendMonoMessageTargetType
 - PhotonNetwork, [128](#)
- SendMonoMessageTargets
 - PhotonNetwork, [128](#)
- SendNext
 - PhotonStream, [148](#)
 - PhotonStreamQueue, [151](#)
- SendOutgoingCommands
 - PhotonNetwork, [123](#)
- sendRate
 - PhotonNetwork, [136](#)
- sendRateOnSerialize
 - PhotonNetwork, [136](#)
- sender
 - PhotonMessageInfo, [100](#)
- SentPickup
 - PickupItem, [170](#)
 - PickupItemSimple, [171](#)
- SequenceChannel
 - RaiseEventOptions, [192](#)
- SerialMatching
 - ExitGames::Client::Photon, [30](#)
- Serialize
 - PhotonStream, [148](#), [149](#)
 - PhotonStreamQueue, [152](#)
- SerializeView
 - PhotonView, [164](#)
- Server
 - PhotonNetwork, [136](#)
- ServerAddress
 - PhotonNetwork, [136](#)
 - ServerSettings, [206](#)
- ServerConnection
 - Enums.cs, [224](#)
- ServerFull
 - ExitGames::Client::Photon::ErrorCode, [43](#)
- ServerPort
 - ServerSettings, [206](#)
- ServerSettings, [204](#)
 - AppId, [206](#)
 - BestRegion, [205](#)
 - DisableAutoOpenWizard, [206](#)
 - EnableLobbyStatistics, [206](#)
 - EnabledRegions, [206](#)
 - HostType, [206](#)
 - HostingOption, [205](#)
 - JoinLobby, [206](#)
 - NotSet, [205](#)
 - OfflineMode, [205](#)
 - PhotonCloud, [205](#)
 - PreferredRegion, [206](#)
 - Protocol, [206](#)
 - RpcList, [206](#)
 - SelfHosted, [205](#)
 - ServerAddress, [206](#)
 - ServerPort, [206](#)
 - ToString, [206](#)
 - UseCloud, [206](#)
 - UseCloudBestRegion, [206](#)
 - UseMyServer, [206](#)
- ServerSettingsCleanedWarning
 - PunWizardText, [190](#)
- ServerSettingsInspector, [206](#)
 - IsAppId, [207](#)
 - OnInspectorGUI, [207](#)
 - ProtocolChoices, [207](#)
 - Tcp, [207](#)
 - Udp, [207](#)
- ServerTime, [208](#)
- ServerTimestamp
 - PhotonNetwork, [136](#)
- ServerWeb
 - AccountService, [33](#)

- SetAuthPostData
 - AuthenticationValues, [37](#)
- SetCustomProperties
 - PhotonPlayer, [141](#)
 - Room, [195](#)
- SetLayerSynchronized
 - PhotonAnimatorView, [92](#)
- SetLevelPrefix
 - PhotonNetwork, [123](#)
- SetMasterClient
 - PhotonNetwork, [123](#)
- SetParameterSynchronized
 - PhotonAnimatorView, [93](#)
- SetPlayerCustomProperties
 - PhotonNetwork, [124](#)
- SetProperties
 - ExitGames::Client::Photon::EventCode, [46](#)
 - ExitGames::Client::Photon::OperationCode, [79](#)
- SetPropertiesListedInLobby
 - Room, [195](#)
- SetReceivingEnabled
 - PhotonNetwork, [124](#), [125](#)
- SetScore
 - ScoreExtensions, [204](#)
- SetSendingEnabled
 - PhotonNetwork, [125](#)
- SetSynchronizedValues
 - PhotonTransformView, [153](#)
 - PhotonTransformViewPositionControl, [155](#)
- SetTeam
 - TeamExtensions, [213](#)
- SettingsButton
 - PunWizardText, [190](#)
- SettingsHighlightLabel
 - PunWizardText, [190](#)
- SetupButton
 - PunWizardText, [190](#)
- SetupCompleteInfo
 - PunWizardText, [190](#)
- SetupServerCloudLabel
 - PunWizardText, [190](#)
- SetupWizardInfo
 - PunWizardText, [190](#)
- SetupWizardTitle
 - PunWizardText, [190](#)
- SetupWizardWarningMessage
 - PunWizardText, [190](#)
- SetupWizardWarningTitle
 - PunWizardText, [190](#)
- ShouldExecute
 - ExitGames::Client::DemoParticle::TimeKeeper, [215](#)
- showGui
 - OnClickInstantiate, [74](#)
- ShowInfoOfPlayer, [208](#)
 - CharacterSize, [208](#)
 - DisableOnOwnObjects, [208](#)
 - font, [208](#)
- ShowRegistrationWizard
 - PhotonEditor, [95](#)
- ShowStatusWhenConnecting, [208](#)
 - Skin, [209](#)
- Skin
 - ShowStatusWhenConnecting, [209](#)
- SkipButton
 - PunWizardText, [190](#)
- SkipRPCListUpdateLabel
 - PunWizardText, [191](#)
- SkipRegistrationInfo
 - PunWizardText, [190](#)
- SliceIncreaseIndex
 - ExitGames::Client::Photon, [29](#)
- SlicePurgeIndex
 - ExitGames::Client::Photon, [29](#)
- SlicePurgeUpToIndex
 - ExitGames::Client::Photon, [29](#)
- SliceSetIndex
 - ExitGames::Client::Photon, [29](#)
- SmoothSyncMovement, [209](#)
 - Awake, [209](#)
 - OnPhotonSerializeView, [209](#)
 - SmoothingDelay, [209](#)
 - Update, [209](#)
- SmoothingDelay
 - SmoothSyncMovement, [209](#)
- SpawnPosition
 - OnJoinedInstantiate, [76](#)
- Sphere
 - ExitGames::Client::GUI, [28](#)
- SqlLobby
 - LoadbalancingPeer.cs, [228](#)
- SqlLobbyFilter
 - ExitGames::Client::Photon::LoadbalancingPeer::↵
OpJoinRandomRoomParams, [80](#)
- Start
 - ConnectAndJoinRandom, [39](#)
 - InRoomChat, [56](#)
 - MoveByKeys, [71](#)
 - PhotonLagSimulationGui, [99](#)
 - PhotonStatsGui, [145](#)
 - PunTeams, [186](#)
 - SupportLogger, [210](#)
 - SupportLogging, [211](#)
- StartButton
 - PunWizardText, [191](#)
- StartPing
 - PingMonoEditor, [173](#)
- StartTime
 - InRoomRoundTimer, [58](#)
- starting_severtime
 - InRoomRoundTimer, [58](#)
- statsOn
 - PhotonStatsGui, [146](#)
- statsRect
 - PhotonStatsGui, [146](#)
- statsWindowOn

- PhotonStatsGui, 146
- Steam
 - LoadbalancingPeer.cs, 227
- StripKeysWithNullValues
 - Extensions, 48
- StripToStringKeys
 - Extensions, 49
- SupportClass
 - Extensions.cs, 225
- SupportLogger, 210
 - LogTrafficStats, 210
 - Start, 210
- SupportLogging, 210
 - LogStats, 211
 - LogTrafficStats, 211
 - OnApplicationPause, 211
 - OnApplicationQuit, 211
 - OnConnectedToPhoton, 211
 - OnCreatedRoom, 211
 - OnDisconnectedFromPhoton, 211
 - OnFailedToConnectToPhoton, 211
 - OnJoinedLobby, 211
 - OnJoinedRoom, 211
 - OnLeftRoom, 211
 - Start, 211
- SuppressRoomEvents
 - ExitGames::Client::Photon::ParameterCode, 89
- suppressRoomEvents
 - RoomOptions, 202
- SwitchToProtocol
 - PhotonNetwork, 125
- synchronization
 - PhotonView, 165
- SynchronizeEnabled
 - PhotonTransformViewPositionModel, 157
 - PhotonTransformViewRotationModel, 159
 - PhotonTransformViewScaleModel, 160
- SynchronizeType
 - PhotonAnimatorView, 91
 - PhotonAnimatorView::SynchronizedLayer, 212
 - PhotonAnimatorView::SynchronizedParameter, 212
- SynchronizeValues
 - PhotonTransformViewPositionModel, 156
- TagObject
 - PhotonPlayer, 142
- Takeover
 - PhotonView.cs, 233
- TargetActorNr
 - ExitGames::Client::Photon::ParameterCode, 89
- TargetActors
 - RaiseEventOptions, 192
- Tcp
 - ServerSettingsInspector, 207
- Team
 - PunTeams, 185
- TeamExtensions, 212
 - GetTeam, 213
 - SetTeam, 213
- TeamPlayerProp
 - PunTeams, 186
- TeleportEnabled
 - PhotonTransformViewPositionModel, 157
- TeleportIfDistanceGreaterThan
 - PhotonTransformViewPositionModel, 157
- TextPos
 - InRoomRoundTimer, 58
- thrust
 - MoveByKeys, 72
- time
 - PhotonNetwork, 136
- TimeKeeper
 - ExitGames::Client::DemoParticle::TimeKeeper, 214
- TimeOfRespawn
 - PickupItem, 170
- TimeoutDisconnect
 - Public API, 19
- timestamp
 - PhotonMessageInfo, 100
- timetostart
 - InRoomRoundTimer, 58
- ToArray
 - PhotonStream, 149
- ToString
 - AuthenticationValues, 37
 - FriendInfo, 50
 - PhotonMessageInfo, 100
 - PhotonPlayer, 142
 - PhotonView, 164
 - Region, 193
 - Room, 196
 - RoomInfo, 198
 - ServerSettings, 206
 - TypedLobby, 216
 - TypedLobbyInfo, 217
- ToStringFull
 - Extensions, 49
 - PhotonPlayer, 142
 - Room, 196
 - RoomInfo, 198
 - WebRpcResponse, 218
- Token
 - AuthenticationValues, 38
- trafficStatsOn
 - PhotonStatsGui, 146
- TrafficStatsWindow
 - PhotonStatsGui, 145
- TransferOwnership
 - PhotonView, 164
- Trigger
 - PhotonAnimatorView, 91
- Type
 - PhotonAnimatorView::SynchronizedParameter, 212
 - TypedLobby, 216

- TypedLobby, 215
 - Default, 216
 - ExitGames::Client::Photon::LoadbalancingPeer::↔
 - OpJoinRandomRoomParams, 80
 - IsDefault, 216
 - Name, 216
 - ToString, 216
 - Type, 216
 - TypedLobby, 216
- TypedLobbyInfo, 216
 - PlayerCount, 217
 - RoomCount, 217
 - ToString, 217
- UNtoPUNLabel
 - PunWizardText, 191
- Udp
 - ServerSettingsInspector, 207
- UiMainWizard
 - PhotonEditor, 95
- UiSetupApp
 - PhotonEditor, 96
- UnAllocateViewID
 - PhotonNetwork, 126
- Uninitialized
 - Public API, 20
- UnityEngine, 31
- UnityEngine.SceneManagement, 31
- UnityEngine.SceneManagement.SceneManager, 203
- UnityEngine::SceneManagement::SceneManager
 - LoadScene, 203
- Unreliable
 - PhotonView.cs, 233
- unreliableCommandsLimit
 - PhotonNetwork, 137
- UnreliableOnChange
 - PhotonView.cs, 233
- Update
 - ConnectAndJoinRandom, 39
 - PhotonEditor, 96
 - PhotonStatsGui, 145
 - SmoothSyncMovement, 209
- UpdatePosition
 - PhotonTransformViewPositionControl, 155
- UpdateRpcList
 - PhotonEditor, 96
- UpdateTeams
 - PunTeams, 186
- UriPath
 - ExitGames::Client::Photon::ParameterCode, 89
- UrlAccountPage
 - PhotonEditor, 96
- UrlAppIDExplained
 - PhotonEditor, 96
- UrlCloudDashboard
 - PhotonEditor, 96
- UrlCompare
 - PhotonEditor, 96
- UrlDevNet
 - PhotonEditor, 96
- UrlForum
 - PhotonEditor, 96
- UrlFreeLicense
 - PhotonEditor, 96
- UrlHowToSetup
 - PhotonEditor, 96
- us
 - Enums.cs, 224
- UseCloud
 - ServerSettings, 206
- UseCloudBestRegion
 - ServerSettings, 206
- UseMyServer
 - ServerSettings, 206
- UseNative
 - PhotonPingManager, 138
- UsePrefabCache
 - PhotonNetwork, 129
- UseRpcMonoBehaviourCache
 - PhotonNetwork, 129
- UserBlocked
 - ExitGames::Client::Photon::ErrorCode, 43
- UserId
 - AuthenticationValues, 38
 - ExitGames::Client::Photon::ActorProperties, 35
 - ExitGames::Client::Photon::ParameterCode, 89
- Validator
 - AccountService, 34
- Version
 - ConnectAndJoinRandom, 39
- versionPUN
 - PhotonNetwork, 129
- ViewID
 - PickupItem, 170
- viewID
 - PhotonView, 166
- ViewSynchronization
 - PhotonView.cs, 233
- Visible
 - PhotonLagSimulationGui, 99
- visible
 - Room, 197
 - RoomInfo, 200
- visibleField
 - RoomInfo, 199
- WarningPhotonDisconnect
 - PunWizardText, 191
- Web
 - OnClickLoadSomething, 75
- WebRpc
 - ExitGames::Client::Photon::OperationCode, 79
 - PhotonNetwork, 126
- WebRpcParameters
 - ExitGames::Client::Photon::ParameterCode, 89
- WebRpcResponse, 217
 - DebugMessage, 218

- Name, [218](#)
- Parameters, [218](#)
- ReturnCode, [218](#)
- ToStringFull, [218](#)
- WebRpcResponse, [218](#)
- WebRpcReturnCode
 - ExitGames::Client::Photon::ParameterCode, [89](#)
- WebRpcReturnMessage
 - ExitGames::Client::Photon::ParameterCode, [89](#)
- WindowId
 - PhotonLagSimulationGui, [99](#)
 - PhotonStatsGui, [146](#)
- WindowRect
 - PhotonLagSimulationGui, [99](#)
- WindowTitle
 - PunWizardText, [191](#)
- WindowType
 - PhotonEditor, [96](#)
- WireCube
 - ExitGames::Client::GUI, [28](#)
- WireSphere
 - ExitGames::Client::GUI, [28](#)
- WizardMainWindowInfo
 - PunWizardText, [191](#)