1. Lua API Reference . . . 20
1.1 Misc . . . 20
1.1.1 Const . . . 20
1.1.2 CreateTable . . . 21
1.1.3 DebugGetProfileTime . . . 21
1.1.4 GetDeltaTime . . . 22
1.1.5 GetInstanceTime . . . 22
1.1.6 GetThreadId . . . 23
1.1.7 IsPaused . . . 23
1.1.8 KillThread . . . 24
1.1.9 LoadLibrary . . . 25
1.1.10 Pause . . . 25
1.1.11 Resume . . . 26
1.1.12 Sleep . . . 26
1.1.13 SpawnFunction . . . 27
1.1.14 Type . . . 27
1.2 Active . . . 28
1.2.1 Active.GetEntity . . . 28
1.2.2 Active.GetMaxSaveDataSize . . . 29
1.2.3 Active.IsMoving . . . 29
1.2.4 Active.IsSafeVolumeIntersected . . . 30
1.2.5 Active.IsSaveDataAccess . . . 30
1.2.6 Active.IsUpright . . . 31
1.2.7 Active.LoadData . . . 31
1.2.8 Active.SaveData . . . 32
1.3 ActivityChannel . . . 33
1.3.1 ActivityChannel.GetConnectionState . . . 33
1.3.2 ActivityChannel.Initialize . . . 33
1.3.3 ActivityChannel.SetCommunicationPermissions . . . 34
1.3.4 ActivityChannel.SetTitle . . . 35
1.4 Animation . . . 35
1.4.1 Animation.Create . . . 35
1.4.2 Animation.GetBlendState . . . 36
1.4.3 Animation.GetBlendTime . . . 37
1.4.4 Animation.GetBlendWeight . . . 37
1.4.5 Animation.GetDuration . . . 38
1.4.6 Animation.GetName . . . 38
1.4.7 Animation.GetPlaybackRate . . . 39
1.4.8 Animation.GetTime . . . 39
1.4.9 Animation.IsValid . . . 40
1.4.10 Animation.SetBlendWeight . . . 41
1.4.11 Animation.SetLoopCallback . . . 41
1.4.12 Animation.SetPlaybackRate . . . 42
1.4.13 Animation.SetTime . . . 43
1.5 ArcadeGame . . . 43
1.5.1 ArcadeGame.EnableMlaa . . . 43
1.5.2 ArcadeGame.EnableSceneRender . . . 44
1.5.3 ArcadeGame.EnableVsync . . . 45
1.5.4 ArcadeGame.GetScreen . . . 45
1.5.5 ArcadeGame.LeaveGame . . . 46
1.6 AsyncCommand . . . 46
1.6.1 AsyncCommand.Cancel . . . 46
1.6.2 AsyncCommand.Create . . . 47
1.6.3 AsyncCommand.Execute . . . 48
1.6.4 AsyncCommand.GetErrorCode . . . 48
1.6.5 AsyncCommand.GetNpErrorCode . . . 49
1.6.6 AsyncCommand.GetResultCount . . . 50
1.6.7 AsyncCommand.GetResults . . . 50
1.6.8 AsyncCommand.IsFinished . . . 51
1.6.9 AsyncCommand.IsInProgress . . . 52
1.6.10 AsyncCommand.Succeeded . . . 52
1.7 AsyncContext . . . 53
1.7.1 AsyncContext.Create . . . 53
1.8 Atmos . . . 53
1.8.1 Atmos.GetParam . . . 53
1.8.2 Atmos.SetParam . . . 54
1.9 Base64 . . . 55
1.9.1 Base64.Decode . . . 55
1.9.2 Base64.DecodeFromString . . . 55
1.9.3 Base64.Encode . . . 56
1.10 BasicGenx . . . 57
1.10.1 BasicGenx.AddAttribute . . . 57
1.10.2 BasicGenx.AddAttributeInt . . . 58
1.10.3 BasicGenx.AddText . . . 59
1.10.4 BasicGenx.AddTextAsInt . . . 60

1.10.5 BasicGenx.CheckText . . . 60
1.10.6 BasicGenx.Create . . . 61
1.10.7 BasicGenx.EndDocument . . . 62
1.10.8 BasicGenx.EndElement . . . 62
1.10.9 BasicGenx.StartDocument . . . 63
1.10.10 BasicGenx.StartElement . . . 64
1.11 BigInt . . . 65
1.11.1 BigInt.Add . . . 65
1.11.2 BigInt.And . . . 65
1.11.3 BigInt.AsHexString . . . 66
1.11.4 BigInt.Create . . . 66
1.11.5 BigInt.Divide . . . 67
1.11.6 BigInt.GetBit . . . 67
1.11.7 BigInt.Multiply . . . 68
1.11.8 BigInt.Negate . . . 69
1.11.9 BigInt.Not . . . 69
1.11.10 BigInt.Or . . . 70
1.11.11 BigInt.Set . . . 70
1.11.12 BigInt.SetBit . . . 71
1.11.13 BigInt.SetZero . . . 71
1.11.14 BigInt.ShiftLeft . . . 72
1.11.15 BigInt.ShiftRight . . . 72
1.11.16 BigInt.Subtract . . . 73
1.11.17 BigInt.ToNumber . . . 74
1.11.18 BigInt.Xor . . . 74
1.12 Camera . . . 75
1.12.1 Camera.Activate . . . 75
1.12.2 Camera.AttachToParent . . . 75
1.12.3 Camera.Create . . . 76
1.12.4 Camera.Deactivate . . . 76
1.12.5 Camera.EnableAvatarCameraSync . . . 77
1.12.6 Camera.GetInverseViewMatrix . . . 78
1.12.7 Camera.GetLookAtPosition . . . 78
1.12.8 Camera.GetPosition . . . 79
1.12.9 Camera.GetProjectionMatrix . . . 79
1.12.10 Camera.GetViewMatrix . . . 80
1.12.11 Camera.GetViewPortFOV . . . 81
1.12.12 Camera.GetViewPortZFar . . . 81
1.12.13 Camera.GetViewPortZNear . . . 82
1.12.14 Camera.IsActivated . . . 82
1.12.15 Camera.SetLookAtPosition . . . 83
1.12.16 Camera.SetPosition . . . 83
1.12.17 Camera.SetViewPortAspect . . . 84
1.12.18 Camera.SetViewPortFOV . . . 84
1.12.19 Camera.SetViewPortZFar . . . 85
1.12.20 Camera.SetViewPortZNear . . . 85
1.13 Cloth . . . 86
1.13.1 Cloth.SetWindAngle . . . 86
1.14 Clubs . . . 87
1.14.1 Clubs.GetClubCount . . . 87
1.14.2 Clubs.GetClubInfo . . . 87
1.14.3 Clubs.GetClubList . . . 88
1.14.4 Clubs.GetInstanceMemberList . . . 89
1.14.5 Clubs.GetMemberCount . . . 90
1.14.6 Clubs.GetMemberInfo . . . 91
1.14.7 Clubs.GetMemberList . . . 92
1.14.8 Clubs.IsRefreshed . . . 93
1.14.9 Clubs.IsRefreshing . . . 93
1.14.10 Clubs.RequestRefresh . . . 94
1.15 Collision . . . 95
1.15.1 Collision.Create . . . 95
1.15.2 Collision.CreateLinearCast . . . 96
1.15.3 Collision.CreateRaycast . . . 97
1.16 Debug . . . 98
1.16.1 Debug.ArcadeGameGetActive . . . 98
1.16.2 Debug.ArcadeGameLoad . . . 99
1.16.3 Debug.ArcadeGameReset . . . 99
1.16.4 Debug.ArcadeGameSetBorderColor . . . 100
1.16.5 Debug.DebuggerRegister . . . 101
1.16.6 Debug.DebuggerSetupProject . . . 101
1.16.7 Debug.DebuggerUnRegister . . . 102
1.16.8 Debug.DrawBox3d . . . 102
1.16.9 Debug.DrawCone3d . . . 103
1.16.10 Debug.DrawCylinder3d . . . 104
1.16.11 Debug.DrawLine3d . . . 104

1.16.12 Debug.DrawQuad3d . . . 105
1.16.13 Debug.DrawRect . . . 106
1.16.14 Debug.DrawSphere3d . . . 106
1.16.15 Debug.DrawText . . . 107
1.16.16 Debug.DrawTriangle3d . . . 107
1.16.17 Debug.GetDevMode . . . 108
1.16.18 Debug.GetStack . . . 109
1.16.19 Debug.GetThreadIds . . . 109
1.16.20 Debug.GetWorldToScreen . . . 110
1.16.21 Debug.KeyIsHeld . . . 110
1.16.22 Debug.KeyIsPressed . . . 111
1.16.23 Debug.KeyIsReleased . . . 112
1.16.24 Debug.MemGetLargestBlock . . . 113
1.16.25 Debug.MemGetTotal . . . 114
1.16.26 Debug.MemGetUsed . . . 115
1.16.27 Debug.MiniGameGetActive . . . 115
1.16.28 Debug.MiniGameGetClosest . . . 116
1.16.29 Debug.MiniGameReset . . . 116
1.16.30 Debug.MouseGetVelocityX . . . 117
1.16.31 Debug.MouseGetVelocityY . . . 118
1.16.32 Debug.MouseIsButtonHeld . . . 118
1.16.33 Debug.MouseIsButtonPressed . . . 119
1.16.34 Debug.MouseIsButtonReleased . . . 120
1.16.35 Debug.RunScript . . . 120
1.16.36 Debug.SceneIsReady . . . 121
1.16.37 Debug.SceneReset . . . 121
1.16.38 Debug.ScreenSetContent . . . 122
1.16.39 Debug.SetErrorCallback . . . 122
1.16.40 Debug.System . . . 123
1.17 Entity . . . 124
1.17.1 Entity.ApplyAngularImpulse . . . 124
1.17.2 Entity.ApplyImpulse . . . 124
1.17.3 Entity.AttachPersonLabel . . . 125
1.17.4 Entity.AttachToBone . . . 125
1.17.5 Entity.AttachToParent . . . 126
1.17.6 Entity.BlendAnimIn . . . 127
1.17.7 Entity.BlendAnimOut . . . 128
1.17.8 Entity.Copy . . . 129
1.17.9 Entity.Create . . . 129
1.17.10 Entity.CreateForceAction . . . 130
1.17.11 Entity.CreatePointForceAction . . . 131
1.17.12 Entity.CreateTorqueAction . . . 131
1.17.13 Entity.DebugSetAxesVisible . . . 132
1.17.14 Entity.DebugSetCollisionVisible . . . 133
1.17.15 Entity.DebugSetHierarchyVisible . . . 133
1.17.16 Entity.DestroyAction . . . 134
1.17.17 Entity.DetachPersonLabel . . . 135
1.17.18 Entity.EnableAction . . . 135
1.17.19 Entity.EnableCollision . . . 136
1.17.20 Entity.EnableCollisionCallbacks . . . 136
1.17.21 Entity.EnableDynamics . . . 137
1.17.22 Entity.EnableDynamicsAutoDeactivation . . . 138
1.17.23 Entity.EnableInterpolatedMode . . . 139
1.17.24 Entity.EnableKeepUpright . . . 139
1.17.25 Entity.EnableLayerCollision . . . 140
1.17.26 Entity.EvaluateAnims . . . 140
1.17.27 Entity.FindMaterials . . . 141
1.17.28 Entity.GetActiveAnim . . . 142
1.17.29 Entity.GetAngularDamping . . . 143
1.17.30 Entity.GetAngularVelocity . . . 143
1.17.31 Entity.GetBoneCount . . . 144
1.17.32 Entity.GetBoneIndex . . . 144
1.17.33 Entity.GetBoneLocalMatrix . . . 145
1.17.34 Entity.GetBoneWorldMatrix . . . 145
1.17.35 Entity.GetCenterOfMassLocal . . . 146
1.17.36 Entity.GetCenterOfMassWorld . . . 147
1.17.37 Entity.GetChildren . . . 147
1.17.38 Entity.GetCloth . . . 148
1.17.39 Entity.GetCollisionLayer . . . 149
1.17.40 Entity.GetContactCount . . . 149
1.17.41 Entity.GetContactEntity . . . 150
1.17.42 Entity.GetContactId . . . 151
1.17.43 Entity.GetContactPoint . . . 151
1.17.44 Entity.GetContactRelVelocity . . . 152
1.17.45 Entity.GetContactUserId . . . 153

1.17.46 Entity.GetCullDistance . . . 153
1.17.47 Entity.GetGravity . . . 154
1.17.48 Entity.GetIndexOfAnim . . . 155
1.17.49 Entity.GetLinearDamping . . . 155
1.17.50 Entity.GetLinearVelocity . . . 156
1.17.51 Entity.GetLocalBounds . . . 157
1.17.52 Entity.GetLocalMatrix . . . 157
1.17.53 Entity.GetMass . . . 158
1.17.54 Entity.GetNumActiveAnims . . . 158
1.17.55 Entity.GetParent . . . 159
1.17.56 Entity.GetPointVelocity . . . 160
1.17.57 Entity.GetPosition . . . 160
1.17.58 Entity.GetRotationQuat . . . 161
1.17.59 Entity.GetRotationXyz . . . 161
1.17.60 Entity.GetScale . . . 162
1.17.61 Entity.GetSiblings . . . 162
1.17.62 Entity.GetUniqueId . . . 163
1.17.63 Entity.GetWorldBounds . . . 164
1.17.64 Entity.GetWorldMatrix . . . 164
1.17.65 Entity.HasActiveAnims . . . 165
1.17.66 Entity.IsActionEnabled . . . 166
1.17.67 Entity.IsDynamicsActive . . . 166
1.17.68 Entity.IsDynamicsAutoDeactivationEnabled . . . 167
1.17.69 Entity.IsInterpolatedModeEnabled . . . 168
1.17.70 Entity.IsKeepUprightEnabled . . . 168
1.17.71 Entity.IsLayerCollisionEnabled . . . 169
1.17.72 Entity.IsVisible . . . 169
1.17.73 Entity.LookAt . . . 170
1.17.74 Entity.LookAtWorld . . . 170
1.17.75 Entity.ParticleIsAlive . . . 171
1.17.76 Entity.ParticleReset . . . 172
1.17.77 Entity.ParticleSetAttachType . . . 172
1.17.78 Entity.ParticleSetEffect . . . 173
1.17.79 Entity.ParticleSetEmitting . . . 173
1.17.80 Entity.ParticleSetUpdate . . . 174
1.17.81 Entity.ParticleSetUserParam . . . 175
1.17.82 Entity.PauseAnim . . . 175
1.17.83 Entity.PlayAnim . . . 176
1.17.84 Entity.RegisterCollisionCallback . . . 177
1.17.85 Entity.Reset . . . 177
1.17.86 Entity.ResumeAnim . . . 178
1.17.87 Entity.SetActionForce . . . 179
1.17.88 Entity.SetActionPosition . . . 179
1.17.89 Entity.SetActionTorque . . . 180
1.17.90 Entity.SetAngularDamping . . . 181
1.17.91 Entity.SetAngularVelocity . . . 181
1.17.92 Entity.SetAttachType . . . 182
1.17.93 Entity.SetCenterOfMassLocal . . . 183
1.17.94 Entity.SetCollision . . . 183
1.17.95 Entity.SetCollisionLayer . . . 184
1.17.96 Entity.SetCullDistance . . . 185
1.17.97 Entity.SetDynamicsActive . . . 185
1.17.98 Entity.SetFrictionCoeff . . . 186
1.17.99 Entity.SetGravity . . . 186
1.17.100 Entity.SetLight . . . 187
1.17.101 Entity.SetLinearDamping . . . 187
1.17.102 Entity.SetLinearVelocity . . . 188
1.17.103 Entity.SetLocalMatrix . . . 189
1.17.104 Entity.SetMass . . . 189
1.17.105 Entity.SetMaxContacts . . . 190
1.17.106 Entity.SetModel . . . 191
1.17.107 Entity.SetPosition . . . 191
1.17.108 Entity.SetRestitutionCoeff . . . 192
1.17.109 Entity.SetRotationQuat . . . 193
1.17.110 Entity.SetRotationX . . . 193
1.17.111 Entity.SetRotationXyz . . . 194
1.17.112 Entity.SetRotationY . . . 194
1.17.113 Entity.SetRotationZ . . . 195
1.17.114 Entity.SetScale . . . 195
1.17.115 Entity.SetScreen . . . 196
1.17.116 Entity.SetSkeleton . . . 196
1.17.117 Entity.SetVisible . . . 197
1.17.118 Entity.StopAnim . . . 198
1.18 Environment . . . 198
1.18.1 Environment.DebugProfileBegin . . . 198

1.18.2 Environment.DebugProfileEnd . . . 199
1.18.3 Environment.EnableCallbackGc . . . 200
1.18.4 Environment.EnableFrameGc . . . 200
1.18.5 Environment.SetCallbackGcInterval . . . 201
1.18.6 Environment.SetCallbackGcStep . . . 202
1.18.7 Environment.SetFrameGcStep . . . 202
1.19 Font . . . 203
1.19.1 Font.Create . . . 203
1.20 Furniture . . . 203
1.20.1 Furniture.ClearLayout . . . 203
1.20.2 Furniture.ExportLayout . . . 204
1.20.3 Furniture.GetCategories . . . 205
1.20.4 Furniture.GetItemCount . . . 205
1.20.5 Furniture.GetItemData . . . 206
1.20.6 Furniture.ImportLayout . . . 207
1.20.7 Furniture.IsSystemBusy . . . 207
1.20.8 Furniture.LoadLayout . . . 208
1.20.9 Furniture.SaveLayout . . . 209
1.20.10 Furniture.SetCallback . . . 209
1.21 GameInfo . . . 210
1.21.1 GameInfo.GetSearchResults . . . 210
1.21.2 GameInfo.IsGamesListBusy . . . 211
1.21.3 GameInfo.IsSearchFinished . . . 212
1.21.4 GameInfo.SearchByTitleId . . . 213
1.22 GameLaunch . . . 214
1.22.1 GameLaunch.CreateSession . . . 214
1.22.2 GameLaunch.GetCurrentFocusPersonId . . . 214
1.22.3 GameLaunch.GetMaxPlayers . . . 215
1.22.4 GameLaunch.GetNumberOfPlayers . . . 215
1.22.5 GameLaunch.GetSessionMasterPersonId . . . 216
1.22.6 GameLaunch.GetTitle . . . 217
1.22.7 GameLaunch.GetUserComment . . . 217
1.22.8 GameLaunch.InitiateLaunch . . . 218
1.22.9 GameLaunch.LeaveSession . . . 218
1.22.10 GameLaunch.OnHideUI . . . 219
1.22.11 GameLaunch.OnShowUI . . . 219
1.22.12 GameLaunch.ReturnJoinGameResult . . . 220
1.22.13 GameLaunch.ReturnLaunchRequestResult . . . 220
1.22.14 GameLaunch.SetMaxPlayers . . . 221
1.22.15 GameLaunch.SetNumPrivateSlots . . . 222
1.22.16 GameLaunch.SetUserComment . . . 222
1.23 GameLaunchButton . . . 223
1.23.1 GameLaunchButton.GetNameRef . . . 223
1.23.2 GameLaunchButton.SetCallbackOnPressed . . . 223
1.23.3 GameLaunchButton.SetText . . . 224
1.24 GameLaunchExporter . . . 225
1.24.1 GameLaunchExporter.BeginSection . . . 225
1.24.2 GameLaunchExporter.BeginSessionExport . . . 225
1.24.3 GameLaunchExporter.EndSection . . . 226
1.24.4 GameLaunchExporter.EndSessionExport . . . 227
1.24.5 GameLaunchExporter.ExportOption . . . 227
1.25 GameLaunchListItem . . . 228
1.25.1 GameLaunchListItem.AddItem . . . 228
1.25.2 GameLaunchListItem.AddItemRaw . . . 228
1.25.3 GameLaunchListItem.GetCallbackFocusItem . . . 229
1.25.4 GameLaunchListItem.GetFocusItem . . . 230
1.25.5 GameLaunchListItem.GetNameRef . . . 230
1.25.6 GameLaunchListItem.RemoveAllItems . . . 231
1.25.7 GameLaunchListItem.RemoveItem . . . 231
1.25.8 GameLaunchListItem.SetCallbackOnChange . . . 232
1.25.9 GameLaunchListItem.SetFocusItem . . . 232
1.26 GameLaunchMenu . . . 233
1.26.1 GameLaunchMenu.AddItemButton . . . 233
1.26.2 GameLaunchMenu.AddItemList . . . 234
1.26.3 GameLaunchMenu.AddItemSlider . . . 234
1.26.4 GameLaunchMenu.FindItemByNameRef . . . 235
1.26.5 GameLaunchMenu.GetFirstItem . . . 236
1.26.6 GameLaunchMenu.GetId . . . 236
1.26.7 GameLaunchMenu.GetNextItem . . . 237
1.27 GameLaunchMenuSys . . . 238
1.27.1 GameLaunchMenuSys.AddMenu . . . 238
1.27.2 GameLaunchMenuSys.CreateMenu . . . 239
1.27.3 GameLaunchMenuSys.FindMenuById . . . 239
1.27.4 GameLaunchMenuSys.GetFirstMenu . . . 240
1.27.5 GameLaunchMenuSys.GetNextMenu . . . 240

1.27.6 GameLaunchMenuSys.RemoveMenu . . . 241
1.28 GameLaunchSlider . . . 242
1.28.1 GameLaunchSlider.GetNameRef . . . 242
1.28.2 GameLaunchSlider.GetValue . . . 243
1.28.3 GameLaunchSlider.SetIntegerSlider . . . 243
1.28.4 GameLaunchSlider.SetPercentageSlider . . . 244
1.28.5 GameLaunchSlider.SetValue . . . 244
1.29 GfxEffect . . . 245
1.29.1 GfxEffect.Create . . . 245
1.29.2 GfxEffect.Enable . . . 245
1.29.3 GfxEffect.IsEnabled . . . 246
1.29.4 GfxEffect.SetAlphaRef . . . 246
1.29.5 GfxEffect.SetBlendMode . . . 247
1.29.6 GfxEffect.SetBounds . . . 248
1.29.7 GfxEffect.SetDepthPrepass . . . 248
1.29.8 GfxEffect.SetDrawRange . . . 249
1.29.9 GfxEffect.SetIndexData . . . 249
1.29.10 GfxEffect.SetRenderStage . . . 250
1.29.11 GfxEffect.SetShadowEnable . . . 251
1.29.12 GfxEffect.SetTexture . . . 251
1.29.13 GfxEffect.SetTransform . . . 252
1.29.14 GfxEffect.SetTwoSided . . . 252
1.29.15 GfxEffect.SetVertexData . . . 253
1.30 Group . . . 253
1.30.1 Group.Create . . . 253
1.30.2 Group.GetId . . . 254
1.30.3 Group.GetMemberCount . . . 255
1.30.4 Group.GetMemberList . . . 255
1.30.5 Group.IsInGroup . . . 256
1.30.6 Group.Join . . . 257
1.30.7 Group.Leave . . . 258
1.31 GroupDoor . . . 258
1.31.1 GroupDoor.Create . . . 258
1.31.2 GroupDoor.Enter . . . 259
1.31.3 GroupDoor.GetFailedEnterMessage . . . 260
1.32 Hash . . . 260
1.32.1 Hash.Create . . . 260
1.32.2 Hash.Finalize . . . 261
1.32.3 Hash.Start . . . 261
1.32.4 Hash.Update . . . 262
1.33 HelpSystem . . . 262
1.33.1 HelpSystem.GetObjectHelpText . . . 262
1.34 HttpPostData . . . 263
1.34.1 HttpPostData.AddDataHeader . . . 263
1.34.2 HttpPostData.AddHttpHeader . . . 263
1.34.3 HttpPostData.AddValue . . . 264
1.34.4 HttpPostData.Create . . . 265
1.34.5 HttpPostData.Finalize . . . 265
1.34.6 HttpPostData.FinalizeData . . . 266
1.34.7 HttpPostData.FinalizeHttpHeaders . . . 266
1.34.8 HttpPostData.Reset . . . 267
1.35 Intersect . . . 267
1.35.1 Intersect.AabbAabb2d . . . 267
1.35.2 Intersect.AabbAabb3d . . . 268
1.35.3 Intersect.AabbPlane3d . . . 269
1.35.4 Intersect.CircleAabb2d . . . 269
1.35.5 Intersect.CircleCircle2d . . . 270
1.35.6 Intersect.DistSqLineLine3d . . . 271
1.35.7 Intersect.LineAabb2d . . . 272
1.35.8 Intersect.LineAabb3d . . . 272
1.35.9 Intersect.LineCircle2d . . . 273
1.35.10 Intersect.LineLine2d . . . 274
1.35.11 Intersect.LinePlane3d . . . 275
1.35.12 Intersect.LineSegment2d . . . 276
1.35.13 Intersect.LineSphere3d . . . 276
1.35.14 Intersect.PlanePlane3d . . . 277
1.35.15 Intersect.RayAabb2d . . . 278
1.35.16 Intersect.RayAabb3d . . . 279
1.35.17 Intersect.RayCircle2d . . . 280
1.35.18 Intersect.RayLine2d . . . 280
1.35.19 Intersect.RayPlane3d . . . 281
1.35.20 Intersect.RayRay2d . . . 282
1.35.21 Intersect.RaySegment2d . . . 283
1.35.22 Intersect.RaySphere3d . . . 284
1.35.23 Intersect.SegmentAabb2d . . . 285

1.35.24 Intersect.SegmentAabb3d . . . 286
1.35.25 Intersect.SegmentCircle2d . . . 287
1.35.26 Intersect.SegmentPlane3d . . . 287
1.35.27 Intersect.SegmentSegment2d . . . 288
1.35.28 Intersect.SegmentSphere3d . . . 289
1.35.29 Intersect.SphereAabb3d . . . 290
1.35.30 Intersect.SpherePlane3d . . . 290
1.35.31 Intersect.SphereSphere3d . . . 291
1.36 Json . . . 292
1.36.1 Json.Decode . . . 292
1.36.2 Json.DecodeFromString . . . 293
1.36.3 Json.Encode . . . 293
1.37 Light . . . 294
1.37.1 Light.Create . . . 294
1.37.2 Light.SetAttenuationEnd . . . 295
1.37.3 Light.SetAttenuationPower . . . 295
1.37.4 Light.SetAttenuationStart . . . 296
1.37.5 Light.SetColor . . . 296
1.37.6 Light.SetMask . . . 297
1.37.7 Light.SetShadowQuality . . . 298
1.37.8 Light.SetSpotFov . . . 298
1.37.9 Light.SetSpotShadow . . . 299
1.38 LinearCast . . . 300
1.38.1 LinearCast.DebugDraw . . . 300
1.38.2 LinearCast.GetEndPoint . . . 300
1.38.3 LinearCast.GetHitCount . . . 301
1.38.4 LinearCast.GetHitData . . . 302
1.38.5 LinearCast.GetHitDistance . . . 303
1.38.6 LinearCast.GetHitNormal . . . 304
1.38.7 LinearCast.GetHitPoint . . . 306
1.38.8 LinearCast.GetHitType . . . 307
1.38.9 LinearCast.GetStartPoint . . . 308
1.38.10 LinearCast.IsReady . . . 309
1.38.11 LinearCast.SetCollisionLayersToHit . . . 310
1.38.12 LinearCast.SetRotation . . . 311
1.38.13 LinearCast.Start . . . 312
1.39 LocalPlayer . . . 313
1.39.1 LocalPlayer.AddRepertoire . . . 313
1.39.2 LocalPlayer.AddToPlayersMetList . . . 314
1.39.3 LocalPlayer.DebugRemovePortable . . . 314
1.39.4 LocalPlayer.DebugSetPortable . . . 315
1.39.5 LocalPlayer.DoAction . . . 315
1.39.6 LocalPlayer.EnableAutoHiding . . . 316
1.39.7 LocalPlayer.GetAccountRegion . . . 317
1.39.8 LocalPlayer.GetAccountTerritory . . . 317
1.39.9 LocalPlayer.GetAge . . . 318
1.39.10 LocalPlayer.GetAvatarRig . . . 318
1.39.11 LocalPlayer.GetCurrentLogin . . . 319
1.39.12 LocalPlayer.GetCurrentRegion . . . 320
1.39.13 LocalPlayer.GetFirstLogin . . . 320
1.39.14 LocalPlayer.GetFriendCount . . . 321
1.39.15 LocalPlayer.GetFriends . . . 322
1.39.16 LocalPlayer.GetHomeAccountId . . . 322
1.39.17 LocalPlayer.GetHomeRegion . . . 323
1.39.18 LocalPlayer.GetLastLogin . . . 323
1.39.19 LocalPlayer.GetLoginCount . . . 324
1.39.20 LocalPlayer.GetPerson . . . 325
1.39.21 LocalPlayer.GetPersonId . . . 325
1.39.22 LocalPlayer.GetPreviousUserType . . . 326
1.39.23 LocalPlayer.GetUser . . . 326
1.39.24 LocalPlayer.GetUserType . . . 327
1.39.25 LocalPlayer.IsChatAllowed . . . 328
1.39.26 LocalPlayer.IsDoingAction . . . 328
1.39.27 LocalPlayer.IsInWardrobe . . . 329
1.39.28 LocalPlayer.Relocate . . . 329
1.39.29 LocalPlayer.RelocateToGroupInstance . . . 330
1.39.30 LocalPlayer.RelocateToUniqueInstance . . . 331
1.39.31 LocalPlayer.RemoveRepertoire . . . 331
1.39.32 LocalPlayer.RestoreAvatar . . . 332
1.39.33 LocalPlayer.SetAvatarRig . . . 332
1.39.34 LocalPlayer.SetClothingItems . . . 333
1.39.35 LocalPlayer.SetRelocateObject . . . 333
1.39.36 LocalPlayer.SetRigComponentObjectId . . . 334
1.40 Material . . . 335
1.40.1 Material.GetParam . . . 335

1.40.2 Material.SetAlphaRef . . . 335
1.40.3 Material.SetBlendMode . . . 336
1.40.4 Material.SetDepthPrepass . . . 337
1.40.5 Material.SetMatrix . . . 337
1.40.6 Material.SetTexture . . . 338
1.40.7 Material.SetVector . . . 339
1.41 Matrix44 . . . 340
1.41.1 Matrix44.Add . . . 340
1.41.2 Matrix44.Copy . . . 341
1.41.3 Matrix44.Create . . . 341
1.41.4 Matrix44.GetRotationXyz . . . 342
1.41.5 Matrix44.GetRow . . . 342
1.41.6 Matrix44.Invert . . . 343
1.41.7 Matrix44.InvertAffine . . . 343
1.41.8 Matrix44.InvertAffineOrthogonal . . . 344
1.41.9 Matrix44.IsEqual . . . 345
1.41.10 Matrix44.IsIdentity . . . 345
1.41.11 Matrix44.Multiply . . . 346
1.41.12 Matrix44.Orthonormalize . . . 346
1.41.13 Matrix44.SetAxisRotation . . . 347
1.41.14 Matrix44.SetIdentity . . . 348
1.41.15 Matrix44.SetRotationQuat . . . 348
1.41.16 Matrix44.SetRotationXyz . . . 349
1.41.17 Matrix44.SetRow . . . 349
1.41.18 Matrix44.SetScale . . . 350
1.41.19 Matrix44.SetTranslation . . . 351
1.41.20 Matrix44.Subtract . . . 351
1.41.21 Matrix44.Transpose . . . 352
1.42 MediaLibrary . . . 353
1.42.1 MediaLibrary.ExportImage . . . 353
1.42.2 MediaLibrary.ExportVideo . . . 353
1.42.3 MediaLibrary.GetSearchResults . . . 354
1.42.4 MediaLibrary.IsExporting . . . 355
1.42.5 MediaLibrary.IsSearchFinished . . . 355
1.42.6 MediaLibrary.IsSearching . . . 356
1.42.7 MediaLibrary.Search . . . 356
1.42.8 MediaLibrary.SetPrepareSearchFiles . . . 357
1.42.9 MediaLibrary.SetPrepareSearchThumbnails . . . 358
1.42.10 MediaLibrary.SetResultsToInclude . . . 358
1.42.11 MediaLibrary.StartSearch . . . 359
1.42.12 MediaLibrary.StopSearch . . . 360
1.43 MemoryContainer . . . 360
1.43.1 MemoryContainer.Create . . . 360
1.43.2 MemoryContainer.DebugLoad . . . 361
1.43.3 MemoryContainer.DebugPrint . . . 361
1.43.4 MemoryContainer.DebugSave . . . 362
1.43.5 MemoryContainer.GetBigInt . . . 363
1.43.6 MemoryContainer.GetBit . . . 363
1.43.7 MemoryContainer.GetBits . . . 364
1.43.8 MemoryContainer.GetFloat16 . . . 364
1.43.9 MemoryContainer.GetFloat32 . . . 365
1.43.10 MemoryContainer.GetInt8 . . . 365
1.43.11 MemoryContainer.GetInt16 . . . 366
1.43.12 MemoryContainer.GetSize . . . 367
1.43.13 MemoryContainer.GetString . . . 367
1.43.14 MemoryContainer.GetUInt8 . . . 368
1.43.15 MemoryContainer.GetUInt16 . . . 368
1.43.16 MemoryContainer.GetUsedSize . . . 369
1.43.17 MemoryContainer.MemCopy . . . 369
1.43.18 MemoryContainer.MemSet . . . 370
1.43.19 MemoryContainer.ReadBigInt . . . 371
1.43.20 MemoryContainer.ReadFloat16 . . . 371
1.43.21 MemoryContainer.ReadFloat32 . . . 372
1.43.22 MemoryContainer.ReadInt8 . . . 372
1.43.23 MemoryContainer.ReadInt16 . . . 373
1.43.24 MemoryContainer.ReadString . . . 373
1.43.25 MemoryContainer.ReadUInt8 . . . 374
1.43.26 MemoryContainer.ReadUInt16 . . . 374
1.43.27 MemoryContainer.Reset . . . 375
1.43.28 MemoryContainer.Seek . . . 376
1.43.29 MemoryContainer.SetBigInt . . . 376
1.43.30 MemoryContainer.SetBit . . . 377
1.43.31 MemoryContainer.SetBits . . . 378
1.43.32 MemoryContainer.SetFloat16 . . . 378
1.43.33 MemoryContainer.SetFloat32 . . . 379

1.43.34 MemoryContainer.SetInt8 . . . 380
1.43.35 MemoryContainer.SetInt16 . . . 380
1.43.36 MemoryContainer.SetString . . . 381
1.43.37 MemoryContainer.SetUsedSize . . . 381
1.43.38 MemoryContainer.Tell . . . 382
1.43.39 MemoryContainer.WriteBigInt . . . 382
1.43.40 MemoryContainer.WriteFloat16 . . . 383
1.43.41 MemoryContainer.WriteFloat32 . . . 384
1.43.42 MemoryContainer.WriteInt8 . . . 384
1.43.43 MemoryContainer.WriteInt16 . . . 385
1.43.44 MemoryContainer.WriteString . . . 385
1.44 MiniGame . . . 386
1.44.1 MiniGame.AddCustomExitMenuOption . . . 386
1.44.2 MiniGame.CloseSession . . . 387
1.44.3 MiniGame.ConfirmReceivedWelcomeMessage . . . 387
1.44.4 MiniGame.DisableWelcomeMessages . . . 388
1.44.5 MiniGame.EnableQueue . . . 388
1.44.6 MiniGame.GetCurrentFocusPersonId . . . 389
1.44.7 MiniGame.GetCurrentFocusPlayerSlot . . . 390
1.44.8 MiniGame.GetJoinType . . . 390
1.44.9 MiniGame.GetMaxPlayerSlots . . . 391
1.44.10 MiniGame.GetNumberOfUsedPlayerSlots . . . 391
1.44.11 MiniGame.GetPersonIdFromPlayerSlot . . . 392
1.44.12 MiniGame.GetPlayerSlotUsedByPersonId . . . 393
1.44.13 MiniGame.GetQueueSize . . . 393
1.44.14 MiniGame.GetTriggerRadius . . . 394
1.44.15 MiniGame.IsAbleToChangeCustomExitMenuOption . . . 394
1.44.16 MiniGame.IsPersonJoined . . . 395
1.44.17 MiniGame.IsSessionOpen . . . 395
1.44.18 MiniGame.IsSessionOwner . . . 396
1.44.19 MiniGame.LocalPlayerLeaveGame . . . 397
1.44.20 MiniGame.OpenSession . . . 397
1.44.21 MiniGame.RemoveCustomExitMenuOption . . . 398
1.44.22 MiniGame.SetJoinType . . . 398
1.44.23 MiniGame.SetLocalPlayerExcludeMessage . . . 399
1.44.24 MiniGame.SetSafeToSendWelcomeMessage . . . 400
1.45 NetPropertyBag . . . 400
1.45.1 NetPropertyBag.Create . . . 400
1.45.2 NetPropertyBag.DeleteFromNet . . . 401
1.45.3 NetPropertyBag.GetBagIndex . . . 402
1.45.4 NetPropertyBag.GetEventBag . . . 402
1.45.5 NetPropertyBag.GetField . . . 403
1.45.6 NetPropertyBag.GetOwnerId . . . 403
1.45.7 NetPropertyBag.GetTypeId . . . 404
1.45.8 NetPropertyBag.GetTypeIdForBag . . . 405
1.45.9 NetPropertyBag.IsOwnedLocally . . . 405
1.45.10 NetPropertyBag.IsReplicated . . . 406
1.45.11 NetPropertyBag.ReplicateToNet . . . 406
1.45.12 NetPropertyBag.SetField . . . 407
1.46 Object . . . 408
1.46.1 Object.DebugCreate . . . 408
1.46.2 Object.DebugDestroy . . . 408
1.46.3 Object.DebugEnableDebugRendering . . . 409
1.46.4 Object.DebugFind . . . 409
1.46.5 Object.DebugGetAllInstanceIds . . . 410
1.46.6 Object.DebugReport . . . 410
1.46.7 Object.DebugRun . . . 411
1.46.8 Object.EnableTargeting . . . 412
1.46.9 Object.GetCurrentListenerData . . . 412
1.46.10 Object.GetInitialPosition . . . 413
1.46.11 Object.GetInitialRotation . . . 413
1.46.12 Object.GetInitialScale . . . 414
1.46.13 Object.GetInstanceId . . . 414
1.46.14 Object.GetInstanceName . . . 415
1.46.15 Object.GetInstanceParameter . . . 416
1.46.16 Object.GetListenerObjects . . . 416
1.46.17 Object.GetLocalizedText . . . 417
1.46.18 Object.GetMe . . . 417
1.46.19 Object.GetNameRef . . . 418
1.46.20 Object.GetObjectId . . . 418
1.46.21 Object.GetPersonId . . . 419
1.46.22 Object.HasFocus . . . 420
1.46.23 Object.HasLocalizedText . . . 421
1.46.24 Object.IsOwnedLocally . . . 421
1.46.25 Object.IsPaused . . . 422

1.46.26 Object.Pause . . . 422
1.46.27 Object.Resume . . . 423
1.46.28 Object.SendListenerData . . . 423
1.46.29 Object.SetFaceActiveTarget . . . 424
1.46.30 Object.SetTargetableLegend . . . 425
1.46.31 Object.SetTargetablePosition . . . 425
1.46.32 Object.SetTargetableText . . . 426
1.47 Osd . . . 426
1.47.1 Osd.GetRoot . . . 426
1.48 OsdBasicAnimGraphic . . . 427
1.48.1 OsdBasicAnimGraphic.AddFrame . . . 427
1.48.2 OsdBasicAnimGraphic.GetNumFrames . . . 427
1.48.3 OsdBasicAnimGraphic.IsAnimating . . . 428
1.48.4 OsdBasicAnimGraphic.SetAnimating . . . 429
1.48.5 OsdBasicAnimGraphic.SetCurrentFrame . . . 429
1.48.6 OsdBasicAnimGraphic.SetFrameRate . . . 430
1.49 OsdBasicChip . . . 430
1.49.1 OsdBasicChip.AddLegendItem . . . 430
1.49.2 OsdBasicChip.CreateContentObject . . . 431
1.49.3 OsdBasicChip.RemoveAllLegendItems . . . 432
1.49.4 OsdBasicChip.SetBannerHeadline . . . 432
1.49.5 OsdBasicChip.SetBannerIconBackTexture . . . 433
1.49.6 OsdBasicChip.SetBannerIconFrontTexture . . . 433
1.49.7 OsdBasicChip.SetBannerIconVisible . . . 434
1.49.8 OsdBasicChip.SetBannerSubheader . . . 434
1.49.9 OsdBasicChip.SetBannerWantVisible . . . 435
1.49.10 OsdBasicChip.SetContentHeight . . . 435
1.49.11 OsdBasicChip.SetContentHeightSetFromChildren . . . 436
1.49.12 OsdBasicChip.SetContentHeightSetFromChip . . . 437
1.49.13 OsdBasicChip.SetContentObjectInFocus . . . 437
1.49.14 OsdBasicChip.SetContentPointInFocus . . . 438
1.49.15 OsdBasicChip.SetContentWidth . . . 438
1.49.16 OsdBasicChip.SetContentWidthSetFromChildren . . . 439
1.49.17 OsdBasicChip.SetContentWidthSetFromChip . . . 439
1.49.18 OsdBasicChip.SetHeightSetFromContent . . . 440
1.49.19 OsdBasicChip.SetSize . . . 441
1.49.20 OsdBasicChip.UpdateContentHeight . . . 441
1.49.21 OsdBasicChip.UpdateContentWidth . . . 442
1.50 OsdBasicLegend . . . 442
1.50.1 OsdBasicLegend.AddItem . . . 442
1.50.2 OsdBasicLegend.BeginAddingItems . . . 443
1.50.3 OsdBasicLegend.EndAddingItems . . . 444
1.50.4 OsdBasicLegend.RemoveAllItems . . . 445
1.50.5 OsdBasicLegend.SetBorderWidth . . . 445
1.50.6 OsdBasicLegend.SetFixedWidth . . . 446
1.50.7 OsdBasicLegend.SetItemSpacing . . . 446
1.50.8 OsdBasicLegend.SetMaxWidth . . . 447
1.50.9 OsdBasicLegend.SetMinWidth . . . 448
1.50.10 OsdBasicLegend.SetText . . . 448
1.51 OsdBasicMenu . . . 449
1.51.1 OsdBasicMenu.AddItem . . . 449
1.51.2 OsdBasicMenu.BeginChanges . . . 449
1.51.3 OsdBasicMenu.EndChanges . . . 450
1.51.4 OsdBasicMenu.GetCurrentItemId . . . 451
1.51.5 OsdBasicMenu.GetItemById . . . 451
1.51.6 OsdBasicMenu.IsCurrentItemSet . . . 452
1.51.7 OsdBasicMenu.OnInputDown . . . 453
1.51.8 OsdBasicMenu.OnInputUp . . . 453
1.51.9 OsdBasicMenu.Reset . . . 454
1.51.10 OsdBasicMenu.SetCurrentItem . . . 455
1.51.11 OsdBasicMenu.SetItemEnabled . . . 455
1.51.12 OsdBasicMenu.SetItemSpacing . . . 456
1.51.13 OsdBasicMenu.SetOnSetItemEnabledCallback . . . 457
1.51.14 OsdBasicMenu.SetOnSetItemSelectedCallback . . . 457
1.52 OsdBasicPopup . . . 458
1.52.1 OsdBasicPopup.SetMessage . . . 458
1.52.2 OsdBasicPopup.SetTitle . . . 459
1.53 OsdBasicSingleGraphic . . . 459
1.53.1 OsdBasicSingleGraphic.SetBackColorType . . . 460
1.53.2 OsdBasicSingleGraphic.SetBackTexture . . . 460
1.53.3 OsdBasicSingleGraphic.SetFrontColorType . . . 461
1.53.4 OsdBasicSingleGraphic.SetFrontTexture . . . 461
1.54 OsdBasicText . . . 462
1.54.1 OsdBasicText.GetText . . . 462
1.54.2 OsdBasicText.SetClipping . . . 463

1.54.3 OsdBasicText.SetFont . . . 463
1.54.4 OsdBasicText.SetIgnoreSpecialCharacters . . . 464
1.54.5 OsdBasicText.SetScrolling . . . 464
1.54.6 OsdBasicText.SetText . . . 465
1.55 OsdObject . . . 466
1.55.1 OsdObject.Activate . . . 466
1.55.2 OsdObject.CreateChildObject . . . 466
1.55.3 OsdObject.Deactivate . . . 467
1.55.4 OsdObject.DestroyObject . . . 468
1.55.5 OsdObject.FindObject . . . 468
1.55.6 OsdObject.GetColorValue . . . 469
1.55.7 OsdObject.GetHeight . . . 469
1.55.8 OsdObject.GetPositionX . . . 470
1.55.9 OsdObject.GetPositionY . . . 470
1.55.10 OsdObject.GetUniqueId . . . 471
1.55.11 OsdObject.GetWidth . . . 471
1.55.12 OsdObject.HandleInput . . . 472
1.55.13 OsdObject.IsActive . . . 473
1.55.14 OsdObject.SetAlphaType . . . 473
1.55.15 OsdObject.SetAlphaValue . . . 474
1.55.16 OsdObject.SetAngle . . . 474
1.55.17 OsdObject.SetColorType . . . 475
1.55.18 OsdObject.SetColorValue . . . 476
1.55.19 OsdObject.SetHeight . . . 476
1.55.20 OsdObject.SetHeightFromChildren . . . 477
1.55.21 OsdObject.SetLocalSpaceAlignPoint . . . 477
1.55.22 OsdObject.SetName . . . 478
1.55.23 OsdObject.SetParentSpaceAlignPoint . . . 478
1.55.24 OsdObject.SetPosition . . . 479
1.55.25 OsdObject.SetScale . . . 480
1.55.26 OsdObject.SetUniqueId . . . 480
1.55.27 OsdObject.SetVisible . . . 481
1.55.28 OsdObject.SetWidth . . . 481
1.55.29 OsdObject.SetWidthFromChildren . . . 482
1.56 OsdSelectionParent . . . 482
1.56.1 OsdSelectionParent.BeginChanges . . . 483
1.56.2 OsdSelectionParent.EndChanges . . . 483
1.56.3 OsdSelectionParent.GetCurrentSelectionId . . . 484
1.56.4 OsdSelectionParent.SetObjectEnabled . . . 484
1.57 OsdSimpleMenu . . . 485
1.57.1 OsdSimpleMenu.AddItem . . . 485
1.57.2 OsdSimpleMenu.SetItemTitle . . . 485
1.58 OsdSkinnedBox . . . 486
1.58.1 OsdSkinnedBox.ResetTextures . . . 486
1.58.2 OsdSkinnedBox.SetColorGradientEnabled . . . 487
1.58.3 OsdSkinnedBox.SetColorGradientMagnitude . . . 487
1.58.4 OsdSkinnedBox.SetColorGradientReversed . . . 488
1.58.5 OsdSkinnedBox.SetColorGradientSubtractive . . . 489
1.58.6 OsdSkinnedBox.SetTexture . . . 489
1.58.7 OsdSkinnedBox.SetTextures . . . 490
1.59 OsdTextLines . . . 491
1.59.1 OsdTextLines.GetNumTextLines . . . 491
1.59.2 OsdTextLines.SetFont . . . 491
1.59.3 OsdTextLines.SetJustification . . . 492
1.59.4 OsdTextLines.SetLineSpacing . . . 493
1.59.5 OsdTextLines.SetText . . . 493
1.60 OsdTextPanel . . . 494
1.60.1 OsdTextPanel.GetTextBorder . . . 494
1.60.2 OsdTextPanel.GetTextLines . . . 494
1.60.3 OsdTextPanel.IsFitHeightToText . . . 495
1.60.4 OsdTextPanel.IsPanelVisible . . . 495
1.60.5 OsdTextPanel.IsScrollBarVisible . . . 496
1.60.6 OsdTextPanel.SetFitHeightToText . . . 497
1.60.7 OsdTextPanel.SetFont . . . 497
1.60.8 OsdTextPanel.SetJustification . . . 498
1.60.9 OsdTextPanel.SetLineSpacing . . . 498
1.60.10 OsdTextPanel.SetPanelVisible . . . 499
1.60.11 OsdTextPanel.SetScrollBarVisible . . . 499
1.60.12 OsdTextPanel.SetScrollType . . . 500
1.60.13 OsdTextPanel.SetText . . . 500
1.60.14 OsdTextPanel.SetTextBorder . . . 501
1.60.15 OsdTextPanel.SetTextColorType . . . 502
1.61 Osk . . . 502
1.61.1 Osk.Close . . . 502
1.61.2 Osk.GetText . . . 503

1.61.3 Osk.IsAvailable . . . 503
1.61.4 Osk.Open . . . 504
1.62 OutboundMessage . . . 505
1.62.1 OutboundMessage.Create . . . 505
1.62.2 OutboundMessage.GetRemainingBufferSize . . . 505
1.62.3 OutboundMessage.Reset . . . 506
1.62.4 OutboundMessage.Send . . . 506
1.62.5 OutboundMessage.SendUnguaranteed . . . 507
1.62.6 OutboundMessage.WriteBigInt . . . 508
1.62.7 OutboundMessage.WriteFloat . . . 508
1.62.8 OutboundMessage.WriteInt8 . . . 509
1.62.9 OutboundMessage.WriteInt16 . . . 510
1.62.10 OutboundMessage.WriteInt32 . . . 510
1.62.11 OutboundMessage.WriteString . . . 511
1.63 Pad . . . 512
1.63.1 Pad.CreatePads . . . 512
1.63.2 Pad.GetAnalogExtent . . . 512
1.63.3 Pad.GetPad . . . 513
1.63.4 Pad.GetSensorForwards . . . 513
1.63.5 Pad.GetSensorRight . . . 514
1.63.6 Pad.GetSensorUp . . . 515
1.63.7 Pad.IsCircleAccept . . . 515
1.63.8 Pad.IsConnected . . . 516
1.63.9 Pad.IsHeld . . . 516
1.63.10 Pad.Reserve . . . 517
1.63.11 Pad.SetActuator . . . 518
1.63.12 Pad.UnReserve . . . 519
1.63.13 Pad.UnReserveAll . . . 520
1.63.14 Pad.WasAnyJustPressed . . . 520
1.63.15 Pad.WasJustPressed . . . 521
1.63.16 Pad.WasJustReleased . . . 522
1.64 Person . . . 522
1.64.1 Person.AttachToEntity . . . 522
1.64.2 Person.DoAction . . . 523
1.64.3 Person.DoActionBroadcast . . . 524
1.64.4 Person.EnableChatBubbles . . . 525
1.64.5 Person.EnableCollision . . . 525
1.64.6 Person.FindInInstance . . . 526
1.64.7 Person.GetAnimReg . . . 527
1.64.8 Person.GetAvatarStatusType . . . 527
1.64.9 Person.GetBehavior . . . 528
1.64.10 Person.GetBehaviorCategory . . . 528
1.64.11 Person.GetBoneCount . . . 529
1.64.12 Person.GetBoneIndex . . . 530
1.64.13 Person.GetBoneLocalMatrix . . . 530
1.64.14 Person.GetBoneWorldMatrix . . . 531
1.64.15 Person.GetCurrentAction . . . 531
1.64.16 Person.GetGroundUserId . . . 532
1.64.17 Person.GetHeight . . . 532
1.64.18 Person.GetId . . . 533
1.64.19 Person.GetIdsOfFriendsInInstance . . . 534
1.64.20 Person.GetIdsOfRemotePeopleInInstance . . . 534
1.64.21 Person.GetInstanceSessionMasterId . . . 535
1.64.22 Person.GetInventoryItemObjectId . . . 535
1.64.23 Person.GetLocalBounds . . . 536
1.64.24 Person.GetMood . . . 537
1.64.25 Person.GetName . . . 537
1.64.26 Person.GetPosition . . . 538
1.64.27 Person.GetRigComponentObjectId . . . 538
1.64.28 Person.GetRigComponentTypes . . . 539
1.64.29 Person.GetRigField . . . 540
1.64.30 Person.GetRotationY . . . 540
1.64.31 Person.GetUser . . . 541
1.64.32 Person.GetVelocity . . . 541
1.64.33 Person.GetWidth . . . 542
1.64.34 Person.GetWorldBounds . . . 542
1.64.35 Person.HasLock . . . 543
1.64.36 Person.InterpolatePosition . . . 544
1.64.37 Person.IsCollision . . . 544
1.64.38 Person.IsLoaded . . . 545
1.64.39 Person.IsLocked . . . 546
1.64.40 Person.IsOnGround . . . 546
1.64.41 Person.IsRigUsingComponent . . . 547
1.64.42 Person.IsValid . . . 547
1.64.43 Person.IsVisible . . . 548

1.64.44 Person.Lock . . . 549
1.64.45 Person.ResetCurrentStatusText . . . 550
1.64.46 Person.SetAnimReg . . . 550
1.64.47 Person.SetBehavior . . . 551
1.64.48 Person.SetCurrentStatusText . . . 552
1.64.49 Person.SetLabelText . . . 552
1.64.50 Person.SetPosition . . . 553
1.64.51 Person.SetRotationY . . . 553
1.64.52 Person.SetVisible . . . 554
1.64.53 Person.Unlock . . . 554
1.65 Phys3dWorld . . . 555
1.65.1 Phys3dWorld.Create . . . 555
1.65.2 Phys3dWorld.Reset . . . 556
1.65.3 Phys3dWorld.SetEnabled . . . 556
1.66 Quaternion . . . 557
1.66.1 Quaternion.Add . . . 557
1.66.2 Quaternion.Conjugate . . . 558
1.66.3 Quaternion.Copy . . . 558
1.66.4 Quaternion.Create . . . 559
1.66.5 Quaternion.Divide . . . 560
1.66.6 Quaternion.Dot . . . 560
1.66.7 Quaternion.Length . . . 561
1.66.8 Quaternion.Multiply . . . 562
1.66.9 Quaternion.Negate . . . 562
1.66.10 Quaternion.Nlerp . . . 563
1.66.11 Quaternion.Normalize . . . 563
1.66.12 Quaternion.SetAxisAngle . . . 564
1.66.13 Quaternion.SetIdentity . . . 564
1.66.14 Quaternion.SetRotationMatrix . . . 565
1.66.15 Quaternion.SetW . . . 566
1.66.16 Quaternion.SetX . . . 566
1.66.17 Quaternion.SetXyzw . . . 567
1.66.18 Quaternion.SetY . . . 567
1.66.19 Quaternion.SetZ . . . 568
1.66.20 Quaternion.Slerp . . . 569
1.66.21 Quaternion.Subtract . . . 569
1.66.22 Quaternion.ToAxisAngle . . . 570
1.66.23 Quaternion.W . . . 570
1.66.24 Quaternion.X . . . 571
1.66.25 Quaternion.Y . . . 572
1.66.26 Quaternion.Z . . . 572
1.67 Raycast . . . 573
1.67.1 Raycast.GetEndPoint . . . 573
1.67.2 Raycast.GetHitEntity . . . 574
1.67.3 Raycast.GetHitPoint . . . 575
1.67.4 Raycast.GetReflectedVector . . . 576
1.67.5 Raycast.GetStartPoint . . . 577
1.67.6 Raycast.GetSurfaceNormal . . . 578
1.67.7 Raycast.GetSurfaceUserId . . . 579
1.67.8 Raycast.HasHit . . . 580
1.67.9 Raycast.IsReady . . . 581
1.67.10 Raycast.Start . . . 582
1.68 ReceivedMessage . . . 583
1.68.1 ReceivedMessage.GetEventMessage . . . 583
1.68.2 ReceivedMessage.GetRemainingSize . . . 584
1.68.3 ReceivedMessage.GetSenderPersonId . . . 584
1.68.4 ReceivedMessage.GetTotalSize . . . 585
1.68.5 ReceivedMessage.GetType . . . 585
1.68.6 ReceivedMessage.ReadBigInt . . . 586
1.68.7 ReceivedMessage.ReadFloat . . . 587
1.68.8 ReceivedMessage.ReadInt8 . . . 587
1.68.9 ReceivedMessage.ReadInt16 . . . 588
1.68.10 ReceivedMessage.ReadInt32 . . . 589
1.68.11 ReceivedMessage.ReadString . . . 589
1.69 Renderer . . . 590
1.69.1 Renderer.Camera2dGetPosition . . . 590
1.69.2 Renderer.Camera2dGetRotation . . . 591
1.69.3 Renderer.Camera2dGetScale . . . 591
1.69.4 Renderer.Camera2dSetPosition . . . 592
1.69.5 Renderer.Camera2dSetRotation . . . 592
1.69.6 Renderer.Camera2dSetScale . . . 593
1.69.7 Renderer.Create . . . 593
1.69.8 Renderer.DrawCircle2d . . . 594
1.69.9 Renderer.DrawLine2d . . . 595
1.69.10 Renderer.DrawLineList2d . . . 595

1.69.11 Renderer.DrawQuad2d . . . 596
1.69.12 Renderer.DrawQuads2d . . . 596
1.69.13 Renderer.DrawRect2d . . . 597
1.69.14 Renderer.DrawSprite . . . 598
1.69.15 Renderer.DrawSprites . . . 599
1.69.16 Renderer.DrawText2d . . . 599
1.69.17 Renderer.DrawTriList2d . . . 600
1.69.18 Renderer.GetOverlay . . . 601
1.69.19 Renderer.GetPosition . . . 601
1.69.20 Renderer.GetScale . . . 602
1.69.21 Renderer.GetSize . . . 602
1.69.22 Renderer.GetTextSize . . . 603
1.69.23 Renderer.SetBlendMode2d . . . 603
1.69.24 Renderer.SetFont . . . 604
1.69.25 Renderer.SetFontHorzAlignment . . . 605
1.69.26 Renderer.SetFontScale . . . 605
1.69.27 Renderer.SetFontVertAlignment . . . 606
1.69.28 Renderer.SetOverlay . . . 606
1.69.29 Renderer.SetPosition . . . 607
1.69.30 Renderer.SetScale . . . 608
1.69.31 Renderer.SetSize . . . 608
1.69.32 Renderer.SetTarget . . . 609
1.70 Resource . . . 609
1.70.1 Resource.CancelRequest . . . 609
1.70.2 Resource.DebugDumpToTty . . . 610
1.70.3 Resource.DebugSave . . . 611
1.70.4 Resource.Exists . . . 611
1.70.5 Resource.Find . . . 612
1.70.6 Resource.GetData . . . 612
1.70.7 Resource.GetDownloadProgress . . . 613
1.70.8 Resource.GetName . . . 614
1.70.9 Resource.GetStatusCode . . . 614
1.70.10 Resource.GetSystemResource . . . 615
1.70.11 Resource.IsLoaded . . . 616
1.70.12 Resource.IsLoading . . . 616
1.70.13 Resource.Release . . . 617
1.70.14 Resource.Request . . . 618
1.70.15 Resource.RequestSecure . . . 618
1.70.16 Resource.RequestTexture . . . 619
1.70.17 Resource.Run . . . 620
1.70.18 Resource.StartLoading . . . 620
1.71 ResourcePack . . . 621
1.71.1 ResourcePack.DebugGetLoadError . . . 621
1.71.2 ResourcePack.GetLoadProgress . . . 622
1.71.3 ResourcePack.GetLoadState . . . 623
1.71.4 ResourcePack.IsLoading . . . 623
1.71.5 ResourcePack.Load . . . 624
1.71.6 ResourcePack.Unload . . . 625
1.72 Rewards . . . 625
1.72.1 Rewards.AddTicket . . . 625
1.72.2 Rewards.CollectTicket . . . 626
1.72.3 Rewards.DoesHaveTicket . . . 627
1.72.4 Rewards.GetCurrentFocusObjectId . . . 627
1.72.5 Rewards.GetCurrentFocusState . . . 628
1.73 RtGame . . . 629
1.73.1 RtGame.AddCustomExitMenuOption . . . 629
1.73.2 RtGame.CreateGame . . . 629
1.73.3 RtGame.CreateGameObject . . . 630
1.73.4 RtGame.DestroyGameObject . . . 631
1.73.5 RtGame.GetConnectionInfo . . . 631
1.73.6 RtGame.GetMaxPlayers . . . 632
1.73.7 RtGame.GetMemberIdFromUser . . . 633
1.73.8 RtGame.GetNumTokens . . . 633
1.73.9 RtGame.GetOwnerMemberId . . . 634
1.73.10 RtGame.GetPlayer . . . 634
1.73.11 RtGame.GetPlayerCount . . . 635
1.73.12 RtGame.GetRoomId . . . 635
1.73.13 RtGame.GetSentInstanceTime . . . 636
1.73.14 RtGame.GetTokenOwnerMemberId . . . 637
1.73.15 RtGame.GetUserFromMemberId . . . 637
1.73.16 RtGame.IsAbleToChangeCustomExitMenuOption . . . 638
1.73.17 RtGame.IsJoined . . . 638
1.73.18 RtGame.IsSessionOpen . . . 639
1.73.19 RtGame.JoinGame . . . 640
1.73.20 RtGame.Leave . . . 640

1.73.21 RtGame.ReleaseToken . . . 641
1.73.22 RtGame.RemoveCustomExitMenuOption . . . 642
1.73.23 RtGame.ReplicateToNet . . . 642
1.73.24 RtGame.RequestToken . . . 643
1.73.25 RtGame.ReturnJoinGameResult . . . 643
1.73.26 RtGame.SendMessage . . . 644
1.73.27 RtGame.SetSessionClosed . . . 644
1.74 RtGameObject . . . 645
1.74.1 RtGameObject.GetInstanceId . . . 645
1.74.2 RtGameObject.GetName . . . 646
1.74.3 RtGameObject.GetOwnerMemberId . . . 646
1.74.4 RtGameObject.GetOwnershipType . . . 647
1.75 SaveData . . . 648
1.75.1 SaveData.Create . . . 648
1.75.2 SaveData.GetState . . . 648
1.75.3 SaveData.StartLoading . . . 649
1.75.4 SaveData.StartSaving . . . 650
1.76 Scene . . . 650
1.76.1 Scene.FindEntity . . . 650
1.76.2 Scene.FindScreen . . . 651
1.76.3 Scene.FindTriggerVolume . . . 651
1.76.4 Scene.GetGraphicsEngineParam . . . 652
1.76.5 Scene.GetInstanceId . . . 653
1.76.6 Scene.GetLaunchParam . . . 653
1.76.7 Scene.GetLayerCollisionDefault . . . 654
1.76.8 Scene.GetListenerObjects . . . 654
1.76.9 Scene.GetMaxSaveDataSize . . . 655
1.76.10 Scene.GetSaveDataPermission . . . 656
1.76.11 Scene.GetStartupInstanceParam . . . 656
1.76.12 Scene.GetStartupValueParam . . . 657
1.76.13 Scene.GetVirtualObject . . . 658
1.76.14 Scene.IsLoading . . . 658
1.76.15 Scene.IsLoadingObjects . . . 659
1.76.16 Scene.IsSaveDataBusy . . . 659
1.76.17 Scene.LoadData . . . 660
1.76.18 Scene.LockPadControls . . . 661
1.76.19 Scene.OverrideAvatarCloseUpCamAspect . . . 661
1.76.20 Scene.OverrideAvatarCloseUpCamFov . . . 662
1.76.21 Scene.OverrideAvatarCloseUpCamZFar . . . 663
1.76.22 Scene.OverrideAvatarCloseUpCamZNear . . . 663
1.76.23 Scene.OverrideAvatarStandardCamAspect . . . 664
1.76.24 Scene.OverrideAvatarStandardCamFov . . . 665
1.76.25 Scene.OverrideAvatarStandardCamZFar . . . 665
1.76.26 Scene.OverrideAvatarStandardCamZNear . . . 666
1.76.27 Scene.ResetAvatarCamOverrides . . . 667
1.76.28 Scene.SaveData . . . 667
1.76.29 Scene.SetGraphicsEngineParam . . . 668
1.76.30 Scene.SetLayerCollisionDefault . . . 669
1.76.31 Scene.SetLightProbeData . . . 669
1.76.32 Scene.SetLightProbeTransform . . . 670
1.77 SceneDownload . . . 671
1.77.1 SceneDownload.CancelDownload . . . 671
1.77.2 SceneDownload.GetCurrentDownload . . . 671
1.77.3 SceneDownload.GetDownloadProgress . . . 672
1.77.4 SceneDownload.GetDownloadSize . . . 673
1.77.5 SceneDownload.QuerySceneCacheStatus . . . 673
1.77.6 SceneDownload.StartDownload . . . 674
1.78 SceneInfo . . . 674
1.78.1 SceneInfo.GetSceneId . . . 674
1.78.2 SceneInfo.GetSceneName . . . 675
1.78.3 SceneInfo.GetUniqueInstanceId . . . 675
1.79 SceneTransition . . . 676
1.79.1 SceneTransition.ExitScript . . . 676
1.79.2 SceneTransition.GetDestinationScene . . . 677
1.79.3 SceneTransition.GetObject . . . 677
1.79.4 SceneTransition.GetOriginScene . . . 678
1.79.5 SceneTransition.GetSceneChangeProgress . . . 678
1.79.6 SceneTransition.IsSceneChangeComplete . . . 679
1.79.7 SceneTransition.StartSceneChange . . . 679
1.80 Screen . . . 680
1.80.1 Screen.AreSceneSoundsMuted . . . 680
1.80.2 Screen.Create . . . 681
1.80.3 Screen.EnableBackground . . . 681
1.80.4 Screen.GetCenter . . . 682
1.80.5 Screen.GetContentCount . . . 683

1.80.6 Screen.GetCurrentContentIndex . . . 683
1.80.7 Screen.GetCurrentTime . . . 684
1.80.8 Screen.GetMatrix . . . 685
1.80.9 Screen.GetNormal . . . 685
1.80.10 Screen.GetSize . . . 686
1.80.11 Screen.GetTotalTime . . . 687
1.80.12 Screen.GetTriggerPosition . . . 687
1.80.13 Screen.GetTriggerRadius . . . 688
1.80.14 Screen.GetVolume . . . 688
1.80.15 Screen.IsContentPlaying . . . 689
1.80.16 Screen.IsContentReady . . . 690
1.80.17 Screen.IsFogEnabled . . . 690
1.80.18 Screen.IsHDFullScreenModeSet . . . 691
1.80.19 Screen.IsInFullScreenMode . . . 691
1.80.20 Screen.IsProtected . . . 692
1.80.21 Screen.IsVisible . . . 693
1.80.22 Screen.Pause . . . 693
1.80.23 Screen.RestartStream . . . 694
1.80.24 Screen.Resume . . . 694
1.80.25 Screen.SetChannel . . . 695
1.80.26 Screen.SetContent . . . 695
1.80.27 Screen.SetCurrentContentIndex . . . 696
1.80.28 Screen.SetCurrentTime . . . 697
1.80.29 Screen.SetData . . . 697
1.80.30 Screen.SetFogEnabled . . . 698
1.80.31 Screen.SetHDFullScreenMode . . . 699
1.80.32 Screen.SetHighPriority . . . 700
1.80.33 Screen.SetMuteSceneSounds . . . 700
1.80.34 Screen.SetProtected . . . 701
1.80.35 Screen.SetTargetable . . . 701
1.80.36 Screen.SetTriggerPosition . . . 702
1.80.37 Screen.SetTriggerRadius . . . 703
1.80.38 Screen.SetType . . . 703
1.80.39 Screen.SetVisible . . . 704
1.80.40 Screen.SetVolume . . . 704
1.81 Screenshot . . . 705
1.81.1 Screenshot.Capture . . . 705
1.81.2 Screenshot.Create . . . 706
1.81.3 Screenshot.IsInProgress . . . 707
1.81.4 Screenshot.Succeeded . . . 707
1.82 Sound . . . 708
1.82.1 Sound.Create . . . 708
1.82.2 Sound.GetVolume . . . 708
1.82.3 Sound.IsPaused . . . 709
1.82.4 Sound.IsPlaying . . . 709
1.82.5 Sound.IsValid . . . 710
1.82.6 Sound.Pause . . . 711
1.82.7 Sound.Play2d . . . 711
1.82.8 Sound.Play3d . . . 712
1.82.9 Sound.Resume . . . 712
1.82.10 Sound.SetPosition . . . 713
1.82.11 Sound.SetRegister . . . 714
1.82.12 Sound.SetVolume . . . 714
1.82.13 Sound.Stop . . . 715
1.83 SoundBank . . . 715
1.83.1 SoundBank.Find . . . 715
1.83.2 SoundBank.Play2d . . . 716
1.83.3 SoundBank.Play3d . . . 716
1.84 SoundStream . . . 717
1.84.1 SoundStream.Find . . . 717
1.84.2 SoundStream.GetCurrentTime . . . 718
1.84.3 SoundStream.GetTotalTime . . . 718
1.84.4 SoundStream.GetVolume . . . 719
1.84.5 SoundStream.IsPaused . . . 719
1.84.6 SoundStream.IsPlaying . . . 720
1.84.7 SoundStream.Pause . . . 720
1.84.8 SoundStream.Play2d . . . 721
1.84.9 SoundStream.Play3d . . . 721
1.84.10 SoundStream.Resume . . . 722
1.84.11 SoundStream.SetCurrentTime . . . 723
1.84.12 SoundStream.SetPosition . . . 723
1.84.13 SoundStream.SetVolume . . . 724
1.84.14 SoundStream.Stop . . . 724
1.85 Sprite . . . 725
1.85.1 Sprite.Create . . . 725

1.85.2 Sprite.GetColor . . . 725
1.85.3 Sprite.GetElapsedAnimTime . . . 726
1.85.4 Sprite.GetPosition . . . 727
1.85.5 Sprite.GetRotation . . . 727
1.85.6 Sprite.GetScale . . . 728
1.85.7 Sprite.GetTotalAnimTime . . . 728
1.85.8 Sprite.IsAnimPaused . . . 729
1.85.9 Sprite.IsAnimPlaying . . . 730
1.85.10 Sprite.PauseAnim . . . 730
1.85.11 Sprite.PlayAnim . . . 731
1.85.12 Sprite.ResumeAnim . . . 731
1.85.13 Sprite.SetColor . . . 732
1.85.14 Sprite.SetPosition . . . 733
1.85.15 Sprite.SetRotation . . . 733
1.85.16 Sprite.SetScale . . . 734
1.85.17 Sprite.SetTexture . . . 734
1.85.18 Sprite.SetUvs . . . 735
1.85.19 Sprite.StopAnim . . . 736
1.86 SpriteAnim . . . 736
1.86.1 SpriteAnim.AddFrames . . . 736
1.86.2 SpriteAnim.Create . . . 737
1.86.3 SpriteAnim.IsLooping . . . 738
1.86.4 SpriteAnim.SetLooping . . . 738
1.87 StreamOps . . . 739
1.87.1 StreamOps.AddFloat32 . . . 739
1.87.2 StreamOps.BoundsVec4 . . . 740
1.87.3 StreamOps.DivFloat32 . . . 740
1.87.4 StreamOps.Float32ToUInt8 . . . 741
1.87.5 StreamOps.HorizAddFloat32 . . . 742
1.87.6 StreamOps.MulFloat32 . . . 742
1.87.7 StreamOps.ShufFloat32 . . . 743
1.87.8 StreamOps.ShufInt8 . . . 744
1.87.9 StreamOps.SubFloat32 . . . 744
1.88 System . . . 745
1.88.1 System.AddChatLogMessage . . . 745
1.88.2 System.CommercePointIsClosed . . . 746
1.88.3 System.CommercePointOpen . . . 746
1.88.4 System.DrawText2dInScene . . . 748
1.88.5 System.EnableIdleDisconnect . . . 748
1.88.6 System.ExecTitle . . . 749
1.88.7 System.GetChatLogMode . . . 750
1.88.8 System.GetConsoleTestValue . . . 750
1.88.9 System.GetExecTitleErrorCode . . . 751
1.88.10 System.GetNatType . . . 751
1.88.11 System.GetRegionsTerritory . . . 752
1.88.12 System.GetScreenBottomRight . . . 753
1.88.13 System.GetScreenTopLeft . . . 753
1.88.14 System.GetUserValue . . . 754
1.88.15 System.GetVideoUploadResult . . . 754
1.88.16 System.GetVideoUploadStatus . . . 755
1.88.17 System.GetVideoUploadUrl . . . 756
1.88.18 System.IsGameLaunchSessionActive . . . 756
1.88.19 System.IsLookXInverted . . . 757
1.88.20 System.IsLookYInverted . . . 757
1.88.21 System.IsVideoSystemLocked . . . 758
1.88.22 System.IsWideScreen . . . 759
1.88.23 System.PostScore . . . 759
1.88.24 System.ProfanityFilterText . . . 760
1.88.25 System.SendNpMessage . . . 760
1.88.26 System.SetChatLogMode . . . 761
1.88.27 System.UploadVideo . . . 761
1.88.28 System.VideoSystemLock . . . 762
1.88.29 System.VideoSystemUnlock . . . 763
1.88.30 System.WebBrowserIsClosed . . . 764
1.88.31 System.WebBrowserOpen . . . 764
1.88.32 System.WriteScriptedGameLaunchXML . . . 765
1.89 Texture . . . 766
1.89.1 Texture.Find . . . 766
1.89.2 Texture.GetHeight . . . 766
1.89.3 Texture.GetWidth . . . 767
1.90 Time . . . 768
1.90.1 Time.AddDays . . . 768
1.90.2 Time.AddHours . . . 768
1.90.3 Time.AddMinutes . . . 769
1.90.4 Time.AddMonths . . . 770

1.90.5 Time.AddSeconds . . . 770
1.90.6 Time.AddWeeks . . . 771
1.90.7 Time.AddYears . . . 772
1.90.8 Time.ConvertToLocal . . . 772
1.90.9 Time.ConvertToUtc . . . 773
1.90.10 Time.Copy . . . 774
1.90.11 Time.Create . . . 774
1.90.12 Time.GetAsPosixTimestamp . . . 775
1.90.13 Time.GetDate . . . 775
1.90.14 Time.GetDayOfMonth . . . 776
1.90.15 Time.GetDayOfWeek . . . 776
1.90.16 Time.GetDifference . . . 777
1.90.17 Time.GetDifferenceInDays . . . 778
1.90.18 Time.GetDifferenceInHours . . . 779
1.90.19 Time.GetDifferenceInMinutes . . . 779
1.90.20 Time.GetDifferenceInMonths . . . 780
1.90.21 Time.GetDifferenceInSeconds . . . 781
1.90.22 Time.GetDifferenceInWeeks . . . 781
1.90.23 Time.GetDifferenceInYears . . . 782
1.90.24 Time.GetHour . . . 783
1.90.25 Time.GetMinute . . . 783
1.90.26 Time.GetMonth . . . 784
1.90.27 Time.GetSecond . . . 785
1.90.28 Time.GetTime . . . 785
1.90.29 Time.GetYear . . . 786
1.90.30 Time.IsValid . . . 786
1.90.31 Time.SetDayOfMonth . . . 787
1.90.32 Time.SetFromLocal . . . 787
1.90.33 Time.SetFromPosixTimestamp . . . 788
1.90.34 Time.SetFromServer . . . 788
1.90.35 Time.SetHour . . . 789
1.90.36 Time.SetMinute . . . 790
1.90.37 Time.SetMonth . . . 790
1.90.38 Time.SetSecond . . . 791
1.90.39 Time.SetYear . . . 791
1.91 TriggerVolume . . . 792
1.91.1 TriggerVolume.AddCallback . . . 792
1.91.2 TriggerVolume.Enable . . . 793
1.91.3 TriggerVolume.EnableDebugRender . . . 795
1.91.4 TriggerVolume.EnableLayerOverlaps . . . 796
1.91.5 TriggerVolume.GetLocalMatrix . . . 797
1.91.6 TriggerVolume.GetOverlapCount . . . 799
1.91.7 TriggerVolume.GetOverlapData . . . 801
1.91.8 TriggerVolume.GetOverlapType . . . 803
1.91.9 TriggerVolume.GetPosition . . . 805
1.91.10 TriggerVolume.IsEnabled . . . 807
1.91.11 TriggerVolume.IsLayerOverlapsEnabled . . . 809
1.91.12 TriggerVolume.RemoveAllCallbacks . . . 811
1.91.13 TriggerVolume.RemoveCallback . . . 813
1.91.14 TriggerVolume.SetLocalMatrix . . . 815
1.91.15 TriggerVolume.SetPosition . . . 817
1.92 User . . . 819
1.92.1 User.GetName . . . 819
1.92.2 User.GetPerson . . . 820
1.92.3 User.GetSceneOwner . . . 821
1.92.4 User.GoToLocation . . . 821
1.92.5 User.InviteToLocation . . . 822
1.92.6 User.IsFriend . . . 822
1.92.7 User.IsInHome . . . 823
1.92.8 User.IsOnline . . . 824
1.93 Vector4 . . . 824
1.93.1 Vector4.Abs . . . 824
1.93.2 Vector4.Add . . . 825
1.93.3 Vector4.Calculate2dLineIntersection . . . 826
1.93.4 Vector4.Copy . . . 826
1.93.5 Vector4.Create . . . 827
1.93.6 Vector4.CreateBroadcast . . . 828
1.93.7 Vector4.Cross . . . 828
1.93.8 Vector4.Dist2Sqr . . . 829
1.93.9 Vector4.Dist3Sqr . . . 830
1.93.10 Vector4.Dist4Sqr . . . 830
1.93.11 Vector4.Divide . . . 831
1.93.12 Vector4.Dot2 . . . 832
1.93.13 Vector4.Dot3 . . . 832
1.93.14 Vector4.Dot4 . . . 833

1.93.15 Vector4.HorizAdd3 . . . 834
1.93.16 Vector4.HorizAdd4 . . . 834
1.93.17 Vector4.Length2 . . . 835
1.93.18 Vector4.Length2Sqr . . . 836
1.93.19 Vector4.Length3 . . . 836
1.93.20 Vector4.Length3Sqr . . . 837
1.93.21 Vector4.Length4 . . . 838
1.93.22 Vector4.Length4Sqr . . . 839
1.93.23 Vector4.Max . . . 839
1.93.24 Vector4.Min . . . 840
1.93.25 Vector4.Multiply . . . 841
1.93.26 Vector4.MultiplyAdd . . . 842
1.93.27 Vector4.Negate . . . 842
1.93.28 Vector4.Normal2 . . . 843
1.93.29 Vector4.Normal3 . . . 844
1.93.30 Vector4.Normal4 . . . 844
1.93.31 Vector4.Normalize2 . . . 845
1.93.32 Vector4.Normalize3 . . . 846
1.93.33 Vector4.Normalize4 . . . 847
1.93.34 Vector4.Recip . . . 847
1.93.35 Vector4.RecipSqrt . . . 848
1.93.36 Vector4.Rotate . . . 849
1.93.37 Vector4.SetBroadcast . . . 849
1.93.38 Vector4.SetW . . . 850
1.93.39 Vector4.SetX . . . 850
1.93.40 Vector4.SetXyzw . . . 851
1.93.41 Vector4.SetY . . . 852
1.93.42 Vector4.SetZ . . . 852
1.93.43 Vector4.Sign . . . 853
1.93.44 Vector4.Signum . . . 853
1.93.45 Vector4.Sqrt . . . 854
1.93.46 Vector4.Subtract . . . 855
1.93.47 Vector4.Transform . . . 855
1.93.48 Vector4.TransformPoint . . . 856
1.93.49 Vector4.TransformVector . . . 857
1.93.50 Vector4.W . . . 858
1.93.51 Vector4.X . . . 859
1.93.52 Vector4.Y . . . 859
1.93.53 Vector4.Z . . . 860
1.94 View . . . 860
1.94.1 View.CalculatePickRay . . . 860
1.94.2 View.WorldToScreen . . . 861
1.95 Wardrobe . . . 862
1.95.1 Wardrobe.ResetCommercePoints . . . 862
1.95.2 Wardrobe.SetCommercePoint . . . 863
1.96 Xml . . . 863
1.96.1 Xml.Create . . . 863
1.96.2 Xml.FindAttribute . . . 864
1.96.3 Xml.FindElement . . . 864
1.96.4 Xml.FindNextElement . . . 865
1.96.5 Xml.GetAttributeName . . . 866
1.96.6 Xml.GetAttributeValueBool . . . 866
1.96.7 Xml.GetAttributeValueFloat . . . 867
1.96.8 Xml.GetAttributeValueInt . . . 867
1.96.9 Xml.GetAttributeValueText . . . 868
1.96.10 Xml.GetElementName . . . 868
1.96.11 Xml.GetElementValueBool . . . 869
1.96.12 Xml.GetElementValueFloat . . . 869
1.96.13 Xml.GetElementValueInt . . . 870
1.96.14 Xml.GetElementValueText . . . 870
1.96.15 Xml.IntoElement . . . 871
1.96.16 Xml.OutOfElement . . . 872
1.96.17 Xml.SetData . . . 872
1.96.18 Xml.SetDataFromContainer . . . 873

Lua API Reference

Lua API is the full scripting cookbook which serves as reference for all of your library and function needs.

Download the CHM from SCEDev or from the download area of this page (see the paperclip just under the title).

See a list of new and changed functions in What's New in Lua.



The functions will soon be incorporated into the wiki itself - right here in this section! Check the Recent Changes in the HDK Help Wiki for updates.



Errors with CHM

If your CHM doesn't open or display properly, it may be due to Windows security.

The error is caused by a Windows security patch that prevents CHM files from being seen properly displayed initially when downloaded online or shared on a network.

To fix:
1. Right-click on the CHM > Properties.
2. Select 'Unblock.'
3. Open the .CHM.

Collapse all
Expand all Collapse all

Misc

Const

Brief

A variety of useful constant values.

Definition

Arguments

None

Return Values

Description

The available constant values are:
MinNum - The smallest positive normalized number. Note that this is a very small positive number, not the most negative normalized number. For this, use -Const.MaxNum.
MaxNum - The largest positive normalized number.
MaxContigInt - The largest representable contiguous integer, that is the integer x such that x, x - 1, x - 2, ..., 0 are all exactly representable as a Lua numbers, but x + 1 is not.

Examples

See Also

None

CreateTable

Brief

Create an empty Lua table pre-allocated to the specified size.

Definition

```
table CreateTable( number arraySize = 0, number recSize = 0 )
```

Arguments

arraySize - The number of array elements to allocate space for.recSize - The number of records (map elements) to allocate space for.

Return Values

An empty table, with space for arraySize array elements and recSize record (map) elements.

Description

Lua doesn't provide a native function to create a table pre-allocated to a certain size, which means that tables without an initializer list perform O(logN) allocations when inserting N elements. This function will create a table with sufficient space for a requested number of array and record elements. The size of the table will however be 0, as it has no elements (only the memory has been pre-allocated).

Examples

```
local cache = CreateTable( 100 )            -- create an empty table with space for 100 array elements
local dictionary = CreateTable( 0, 50 )     -- create an empty table with space for 50 records
local pointless = CreateTable()             -- default of 0, 0 is same as { }
print(#cache, #dictionary, #pointless)      -- prints "0 0 0"
```

See Also

None

DebugGetProfileTime

Brief

Return a debug timing value.

Definition

```
number DebugGetProfileTime()
```

Arguments

None

Return Values

Current time in seconds.

Description

Returns a timing value in seconds.

Examples

See Also

None

GetDeltaTime

Brief

Returns the time in seconds between this frame and the last frame.

Definition

```
number GetDeltaTime()
```

Arguments

None

Return Values

The time in seconds between this frame and the last frame.

Description

Returns the time in seconds between this frame and the last frame.

Examples

```
print( "Delta time is " .. GetDeltaTime() )
```

See Also

- GetInstanceTime

GetInstanceTime

Brief

Returns the current instance's time in milliseconds.

Definition

```
number, number GetInstanceTime()
```

Arguments

None

Return Values

The current instance time in milliseconds and nil if not in an instance.

Description

This returns the current world instance's time in milliseconds. The instance time starts counting when the instance was created and wraps almost every 50 days. Due to the precision of a lua number the time is split into two values. The bottom value contains 3 bytes and the top half contains the most significant 2 bytes. The majority of applications will not require the top number. The Home client must be run in online mode in order for this function to work as above. If the client is running offline for testing, the time since startup will be returned, in the same format. The following only applies to the online mode. Instance time is not designed for tracking real-world time with 100% accuracy due to latency between clients. It can slow down, speed up and even run backwards and is best used as a mechanism to sync short-term events between clients (e.g. launch a firework in 500ms and explode it 1200ms after that). This creates the effect of events occurring at the same time as far as the player is concerned, regardless of lag. If you need a mechanism to sync over longer periods of time, use Time.SetFromLocal. But note that keeping it updated by adding deltas to it will also cause it to eventually drift out of sync due to a build up of tiny errors. Long term syncing of events between clients should use the NP time with the Lua script checking it once every thirty seconds or so. Once the desired time is near, switch to another timer that is better suited for short time measurements.

Examples

```
b,t = GetInstanceTime()
print( string.format( "The instance time in hex is %#.2x%.6x", t, b ) )
```

See Also

- GetDeltaTime

GetThreadId

Brief

Returns the thread ID for the Lua environment.

Definition

```
number GetThreadId()
```

Arguments

None

Return Values

The thread ID.

Description

Returns the thread ID for the Lua environment.

Examples

```
print( "My id is " .. GetThreadId() )
```

See Also

None

IsPaused

Brief

Returns true if the environment is paused and false otherwise.

Definition

```
boolean IsPaused()
```

Arguments

None

Return Values

true if the environment is paused and false otherwise.

Description

Returns true if the environment is paused and false otherwise.

Examples

```
print( "I am paused = " .. IsPaused() )
```

See Also

- Pause
- Resume

KillThread

Brief

Kills the specified thread.

Definition

```
KillThread( number threadId )
```

Arguments

threadId - The thread to kill.

Return Values

None

Description

Kills the specified thread when the thread has yeilded or teminated.

Examples

```
KillThread( GetThreadId() )
print( "This will still print" )
Sleep( 0.0 )
```

See Also

None

LoadLibrary

Brief

Loads the specified library.

Definition

```
LoadLibrary( string name )
```

Arguments

name - Name of the library to load.

Return Values

None

Description

Loads the specified library. You may not have permision for specific libraries as components can grant library permission.

Examples

```
LoadLibrary( "Vector4" )
```

See Also

None

Pause

Brief

Pauses the Lua environment.

Definition

```
Pause()
```

Arguments

None

Return Values

None

Description

Pauses the Lua environment. Threads are suspended and update functions are not called.

Examples

```
Pause()
```

See Also

- Resume
- IsPaused

Resume

Brief

Resumes a paused Lua environment.

Definition

```
Resume()
```

Arguments

None

Return Values

None

Description

Resumes a paused Lua environment.

Examples

```
Resume()
```

See Also

- Pause

Sleep

Brief

Sleeps the current thread for a specified amount of time.

Definition

```
Sleep( number timeToSleep )
```

Arguments

timeToSleep - The time to sleep in seconds.

Return Values

None

Description

Sleeps the current thread for a specified amount of time. Only works for child threads.

Examples

```
function Hello()
    while true do
        print( "hello" )
        Sleep( 10.0 )
    end
end

SpawnFunction( Hello )
```

See Also

- SpawnFunction

SpawnFunction

Brief

Spawns the specified function in a child Lua thread.

Definition

```
number SpawnFunction( function functionName )
```

Arguments

functionName - The name of the function to spawn.

Return Values

The child thread ID value, or -1 on failure.

Description

Spawns the specified function in a child Lua thread. The function must sleep or yield otherwise it will execute forever.

Examples

```
function Hello()
    while true do
        print( "hello" )
        Sleep( 10.0 )
    end
end

SpawnFunction( Hello )
```

See Also

- Sleep

Type

Brief

Returns the type of the Home specific object.

Definition

```
string Type( userdata object )
```

Arguments

object - A Home specific type.

Return Values

The name of the type.

Description

Returns the type of the Home specific object. Similar to the Lua native type() command, except that where Lua's type() function will only return "userdata" for all API types, the Type function will return a string representing the API object type.

Examples

```
v = Vector4.Create()
print( type( v ) ) -- Prints "userdata"
print( Type( v ) ) -- Prints "Vector4"
```

See Also

None

Active

Active.GetEntity

Brief

Returns the entity for the active item model.

Definition

```
Entity Active.GetEntity()
```

Arguments

None

Return Values

The entity for the active item model.

Description

Returns the entity for the active item model so other entities can attach themselves to it or query it such as its position.

Examples

```
e = Active.GetEntity()
```

See Also

- Entity.GetPosition

Active.GetMaxSaveDataSize

Brief

Returns the maximum size for the save data allowed for this object.

Definition

```
number Active.GetMaxSaveDataSize()
```

Arguments

None

Return Values

The maximum size in bytes.

Description

Gets the maximum size in bytes of the save data for this object. Any attempt to save or read beyond this will generate an error. This value is guaranteed not to decrease between revisions, and will be at least 4096 bytes.

Examples

```
local maxSize = Active.GetMaxSaveDataSize()
```

See Also

None

Active.IsMoving

Brief

Returns true if the active item is currently being moved.

Definition

```
boolean Active.IsMoving()
```

Arguments

None

Return Values

Returns true if the active item is being moved, or false if it has been placed.

Description

Indicates whether the active item is moving or placed.

Examples

```
myActiveAnimationEnabled = Active.IsMoving()
```

See Also

None

Active.IsSafeVolumeIntersected

Brief

Returns true if the active item's safe volume is currently obstructed by another furniture item.

Definition

```
Entity Active.IsSafeVolumeIntersected()
```

Arguments

None

Return Values

True if the active item's safe volume is obstructed, or false if not.

Description

Indicates whether the active item's safe volume is currently intersected by any furniture items.

Examples

```
g_miniGameDisabled = Active.IsSafeVolumeIntersected()
```

See Also

None

Active.IsSaveDataAccess

Brief

Returns true if this instance is allowed to save data and false otherwise.

Definition

```
boolean Active.IsSaveDataAccess()
```

Arguments

None

Return Values

true if this instance is allowed to save data and false otherwise.

Description

The only instance that can save data is the owner of the apartment.

Examples

```
if Active.IsSaveDataAccess() == true then
    mc = MemoryContainer.Create( Active.GetMaxSaveDataSize() )
    Active.LoadData( mc )
end
```

See Also

None

Active.IsUpright

Brief

Returns true if the active item is currently upright.

Definition

```
boolean Active.IsUpright()
```

Arguments

None

Return Values

Returns true if the active item's safe volume is upright, or false if not.

Description

Indicates whether the active item is currently upright, that is, if it is considered to be in the correct orientation to be usable.

Examples

```
g_miniGameDisabled = not Active.IsUpright()
```

See Also

None

Active.LoadData

Brief

Copies the Active's save data into the supplied memory container.

Definition

```
void Active.LoadData(MemoryContainer memContainer, int size = Active.GetMaxSaveDataSize(), int offset
= 0)
```

Arguments

memContainer - The memory container to load into.size - Number of bytes to obtain.offset - Offset in bytes into the memory container at which to store the data.

Return Values

None.

Description

All actives have an allocated save data area which will be persistent across loads of the same object instance. Each instance has a separate save data area. Note that an active should not rely on specific data being present in the save data (as the user may delete their Home save data at any time) and thus actives should be able to recognise and handle the case where no save data is present. When an Active is created all of its save data is set to zero.

Examples

```
local memContainer = MemoryContainer.Create(Active.GetMaxSaveDataSize())
Active.LoadData(memContainer)
if (memContainer:GetUint16(0) == 48879) then        -- 0xBEEF
    print('Save data is valid: found the beef')
    g_mySize = memContainer:GetFloat32(4)
end
```

See Also

None

Active.SaveData

Brief

Copies the data from the supplied memory container into the Active's save data area and requests it to save.

Definition

```
Active.SaveData(MemoryContainer memContainer, number size = Active.GetMaxSaveDataSize(), int offset =
0)
```

Arguments

memContainer - The memory container to obtain the data from.size - Number of bytes to obtain.offset - Offset in bytes into the memory container at which to start reading data.

Return Values

None.

Description

All actives have an allocated save data area which will be persistent across loads of the same object instance. Each instance has a separate save data area. Note that an active should not rely on specific data being present in the save data (as the user may delete their Home save data at any time) and thus actives should be able to recognize and handle the case where no save data is present.
Note: If size is not specified and your memory container is bigger than the maximum save data size an error will occur to catch accidental data truncation.

Examples

```
local memContainer = MemoryContainer.Create(8)
memContainer:SetUint16(0, 48879)        -- 0xBEEF, tags our data as valid
memContainer:SetFloat32(4, mySize)      -- save our current size so we can keep growing next time
Active.SaveData(memContainer, 8)
```

See Also

None

ActivityChannel

ActivityChannel.GetConnectionState

Brief

Used to determine the connection state of the Activity Channel.

Definition

```
ActivityChannelState ActivityChannel.GetConnectionState()
```

Arguments

None

Return Values

The connection state of the Activity Channel.
One of the following status codes:
ActivityChannelState.Disconnected - disconnected from Activity Channel.
ActivityChannelState.Connecting - connecting to Activity Channel.
ActivityChannelState.Connected - connected to Activity Channel.
ActivityChannelState.Disconnecting - disconnecting from Activity Channel.

Description

Sometimes it is useful to know whether the Activity Channel is connected or not, or indeed if the channel is in the process of connecting or disconnecting, e.g. for telling the user when they are able to chat, or ensuring an RTGame can be successfully started (which requires the Activity Channel to be disconnected).

Examples

```
if ActivityChannel.GetConnectionState() == ActivityChannelState.Disconnected then
    StartRTGame()
end
```

See Also

- ActivityChannel.Initialize

ActivityChannel.Initialize

Brief

Used to initialize or shut down the Chat Log Activity Channel.

Definition

```
ActivityChannel.Initialize( boolean value = true )
```

Arguments

value - true to initialize the activity channel, or false stop the activity channel.

Return Values

None.

Description

While players are joined to the game they can communicate privately via the Activity Channel, if initialized with this function. By default, an Activity Channel will not be initialized, and therefore will not be available. In all cases a channel is only created if the minigame/RTGame has a session size of 2 or more, and at least 2 people must be joined to the game in order to initialize the channel. This function should be called during initialization of the mini-game, before joining the game.

Examples

```
ActivityChannel.Initialize()
```

See Also

None

ActivityChannel.SetCommunicationPermissions

Brief

Set communication permissions for who each remote user in the activity channel (text and/or voice).

Definition

```
ActivityChannel.SetCommunicationPermissions( Person otherPerson, boolean isAllowedReceiveTextFrom,
boolean isAllowedReceiveVoiceFrom, boolean isAllowedSendTextTo, boolean isAllowedSendVoiceTo )
```

Arguments

remotePerson - remote person to set activity channel permissions onisAllowedReceiveTextFrom - whether we receive text from this remote personisAllowedReceiveVoiceFrom - whether we receive voice from this remote personisAllowedSendTextTo - currently ignoredisAllowedSendVoiceTo - whether we can send voice to this remote person

Return Values

None.

Description

This allows developers to set communications permissions between the local person and a remote person. Params isAllowedSendTextTo and isAllowedSendVoiceTo are used to determine whether the local user is able to send text or speech to another user, and affect whether a person will be visible in the chat-log. Note though that the client does not currently block outgoing communication on text, only incoming. The permissions are not replicated and so must be done on each client.

Examples

```
if (inHotSeat) then
    ActivityChannel.SetCommunicationPermissions( otherPlayerInQuizGame, false, false, false, false )
end
```

See Also

- ActivityChannel.Initialize

ActivityChannel.SetTitle

Brief

Set a descriptive title to the activity chat channel.

Definition

```
ActivityChannel.SetTitle( string channelTitle = nil )
```

Arguments

channelTitle - New channel title (if null or none supplied will default to default name)

Return Values

None.

Description

Lets people personalize their chat channel. Must be set after initializing the channel.

Examples

```
ActivityChannel.Initialize( );
ActivityChannel.SetTitle( "Team A" );
```

See Also

- ActivityChannel.Initialize

Animation

Animation.Create

Brief

Create an uninitialized animation object.

Definition

```
Animation Animation.Create()
```

Arguments

None

Return Values

An uninitialized animation object.

Description

Create a new uninitialized animation object. Can be passed to Entity.GetActiveAnim() to avoid uncontrolled memory allocation. No other Animation functions are valid on this instance until it is initialized with an active animation.

Examples

```
local anim = Animation.Create()
myEntity:GetActiveAnim(index, anim)
print(anim:GetDuration())
```

See Also

- Entity.GetActiveAnim

Animation.GetBlendState

Brief

Access the current blend state of the active animation.

Definition

```
AnimBlendState Animation.GetBlendState(Animation anim)
```

Arguments

anim - active animation to query

Return Values

An AnimBlendState value:
AnimBlendState.Zero - animation has blended out to zero.
AnimBlendState.In - animation is being blended in.
AnimBlendState.Out - animation is being blended out.
AnimBlendState.Playing - animation is playing at full blend weighting

Description

Returns the current blend state of an active animation

Examples

```
if (anim:GetBlendState() == AnimBlendState.Playing) then
    print("Anim is playing at max blend")
end
```

See Also

- Entity.BlendAnimIn
- Entity.BlendAnimOut
- Animation.GetBlendWeight
- Animation.GetBlendTime

Animation.GetBlendTime

Brief

Access the time within the ease in/out period.

Definition

```
number Animation.GetBlendTime( Animation anim )
```

Arguments

anim - active animation to query

Return Values

The time within the ease in/out period

Description

Access the time within the ease in/out period.

Examples

```
if (anim:GetBlendTime() == 0) then
    print("Blend hasn't started yet")
end
```

See Also

- Entity.BlendAnimIn
- Entity.BlendAnimOut
- Animation.GetBlendWeight
- Animation.GetBlendState

Animation.GetBlendWeight

Brief

Access the current blending weight of the specified active animation.

Definition

```
number Animation.GetBlendWeight(Animation anim)
```

Arguments

anim - active animation to query

Return Values

The current blending weight of the active animation, in the range [0, 1]

Description

Returns the current blending weight of the specified active animation.

Examples

```
local w = anim:GetBlendWeight()
if (w > 0) then
    print("Animation is blended in!")
end
```

See Also

- Entity.BlendAnimIn
- Entity.BlendAnimOut
- Animation.GetBlendTime
- Animation.GetBlendState

Animation.GetDuration

Brief

Get the duration of the animation in seconds.

Definition

```
number Animation.GetDuration( Animation anim )
```

Arguments

anim - A valid active animation.

Return Values

Duration of the animation in seconds.

Description

Get the duration of the animation in seconds.

Examples

```
local duration = anim:GetDuration()
```

See Also

- Animation.GetTime

Animation.GetName

Brief

Access the name of the animation resource being played.

Definition

```
string Animation.GetName( anim )
```

Arguments

anim - the active animation to be queried

Return Values

The name of the animation resource that is being played by the active animation, or nil if no active anim.

Description

Access the name of the animation resource being played by the active animation.

Examples

```
local name = anim:GetName()
```

See Also

None

Animation.GetPlaybackRate

Brief

Returns the current playback rate of the animation

Definition

```
number Animation.GetPlaybackRate(Animation anim)
```

Arguments

anim - the animation of which to query the playback rate

Return Values

the playback rate of the animation

Description

The value returned is a multiple of the normal playback rate, so 1 is normal rate, and -1 is reversed

Examples

```
number animSpeed = Animation.GetPlaybackRate(anim)
```

See Also

- Animation.SetPlaybackRate

Animation.GetTime

Brief

Get the current playback time within a specified animation.

Definition

```
number Animation.GetTime( Animation anim )
```

Arguments

anim - A valid active animation.

Return Values

Current playback time in seconds.

Description

Get the current animation playback time. This is the time index within the animation that the animation is currently located at.

Examples

```
local currAnimTime = anim:GetTime()
if (currAnimTime > 5.0) then
    ChangeAnim()
end
```

See Also

- Animation.SetTime

Animation.IsValid

Brief

Returns true if the animation object is valid.

Definition

```
boolean Animation.IsValid( Animation anim )
```

Arguments

anim - The animation object.

Return Values

True if the animation object is valid, false otherwise.

Description

When using the function variants that modify existing animation objects rather than return new instances, if an object has no active animation to return then the animation object will become invalid to reflect this. A script should check this before using the animation object, similarly to checking for nil when using the original "return new instance" version.

Examples

```
local anim = Animation.Create()
myEntity:GetActiveAnim(index, anim)
if (anim:IsValid()) then
    print("Entity has an active anim")
end
```

See Also

- Animation.Create
- Entity.GetActiveAnim

Animation.SetBlendWeight

Brief

Sets the blend weight of the specified active animation.

Definition

```
Animation.SetBlendWeight(Animation anim, number weight)
```

Arguments

anim - active animation to set weighting of.weight - the desired weighting for the animation (mut be between 0 and 1).

Return Values

None.

Description

If an animation is currently blending in or out, then setting the blend weight directly will stop the automatic blending, and leave the animations weighting to the one set through lua. Animation.GetBlendTime will return 0, and Animation.GetBlendState will return AnimBlendState.Playing. Use this function to manually control the animation blend weights without using the automatic blending.

Examples

```
anim1:SetBlendWeight(0.5)
```

See Also

- Entity.BlendAnimIn
- Entity.BlendAnimOut
- Entity.PlayAnim
- Animation.GetBlendTime
- Animation.GetBlendState

Animation.SetLoopCallback

Brief

Define a Lua function to be called when an animation loops or finishes.

Definition

```
Animation.SetLoopCallback(function callback, userData = nil)
```

Arguments

callback - The Lua function to call. Should be a globally-scoped function.userData - An object to pass to the callback function. Can be any valid Lua or API type.

Return Values

None.

Description

If a script wishes to trigger a new anim or perform other processing as soon as an existing anim finishes or loops, it can register a callback to occur on this event using this function. PS Home will call the specified function at the exact time of finishing or looping, prior to applying the animation to the skeleton. The callbacks are only valid while the Animation object on which they were registered exists; if the Animation object is garbage collected the callbacks will be deactivated. The animation object must be valid in order to set a callback.

Examples

```
entity = Entity.Create()
walkToDoorAnim = Animation.Create()

entity:SetModel("Model")
entity:SetSkeleton("Skeleton")

entity:BlendAnimIn("WalkToDoor", false, 0, AnimBlendType.Linear, 0.2, 0, walkToDoorAnim)

-- set up a callback to occur at end of WalkToDoor, and make it pass us walkToDoorAnim as userdata
walkToDoorAnim:SetLoopCallback(AnimCallback, walkToDoorAnim)

function AnimCallback(anim)
    if (anim == walkToDoorAnim) then
        entity:BlendAnimOut(walkToDoorAnim, AnimBlendType.Linear, 0.1)
        entity:BlendAnimIn("OpenDoor", false, 0, AnimBlendType.Linear, 0.1)
        -- if you want to register a new callback here, make sure the anim object lives beyond the
callback scope
    end
end
```

See Also

- Entity.BlendAnimIn

Animation.SetPlaybackRate

Brief

Sets the playback rate of an animation

Definition

```
Animation.SetPlaybackRate(Animation anim, number pbRate)
```

Arguments

anim - animation on which to set the playback ratepbRate - playback rate to set (can be zero, effectively pausing the anim). The playback rate is a multiplier, so 1 is normal speed, 0.5 is half speed
e, and -1 is reverse, and so on.

Return Values

none

Description

If the playback rate is set such that the animation will loop more than once in a single frame, any loop callbacks will only be called once.

Examples

```
Animation.SetPlaybackRate(reverseAnim, -1)
```

See Also

- Animation.GetPlaybackRate
- Entity.PlayAnim
- Entity.BlendAnimIn

Animation.SetTime

Brief

Set the current playback time within the specified animation.

Definition

```
Animation.SetTime( Animation anim, number time )
```

Arguments

anim - A valid active animation.time - Playback time to set in seconds.

Return Values

None

Description

Set the current animation playback time.

Examples

```
anim:SetTime(0.0)   -- set animation back to beginning
```

See Also

- Animation.GetTime

ArcadeGame

ArcadeGame.EnableMlaa

Brief

Enable or disable the MLAA (morphological anti-aliasing) process.

Definition

```
ArcadeGame.EnableMlaa(boolean enable)
```

Arguments

enable - True to enable or false to disable.

Return Values

None.

Description

PS Home uses the MLAA technique to perform anti-aliasing, and the implementation of this technique introduces one frame of rendering latency into the rendering pipeline. If the arcade game wishes to reduce latency this function can be used to disable the MLAA to remove this additional frame delay. Ensure that you re-enable it upon game exit.

Examples

```
ArcadeGame.EnableMlaa(false)
```

See Also

- ArcadeGame.EnableSceneRender
- ArcadeGame.EnableVsync

ArcadeGame.EnableSceneRender

Brief

Enable or disable the render of the 3D scene.

Definition

```
ArcadeGame.EnableSceneRender(boolean enable)
```

Arguments

enable - True to enable or false to disable.

Return Values

None.

Description

When playing an arcade game the current 3D scene is invisible, so it may be advantageous to a script to disable the render to reclaim some performance for the arcade game. Ensure that you re-enable it upon game exit.

Examples

```
ArcadeGame.EnableSceneRender(false)
```

See Also

- ArcadeGame.EnableMlaa
- ArcadeGame.EnableVsync

ArcadeGame.EnableVsync

Brief

Enable or disable the standard vsync timing.

Definition

```
ArcadeGame.EnableVsync(boolean enable)
```

Arguments

enable - True to enable or false to disable.

Return Values

None.

Description

By default PS Home enables vertical syncing (vsync) which limits the frame rate to 30 frames per second (fps). This function can be used to disable this and allow the arcade game to run at 60 fps, which will make the game feel more reactive and be more suitable for fast "twitch" style games.

Examples

```
ArcadeGame.EnableVsync(false)
```

See Also

- ArcadeGame.EnableSceneRender
- ArcadeGame.EnableMlaa

ArcadeGame.GetScreen

Brief

Returns the Arcade Game's screen.

Definition

```
Screen ArcadeGame.GetScreen()
```

Arguments

None

Return Values

The Arcade Game's screen.

Description

Returns the active Arcade Game's screen.

Examples

```
screen   = ArcadeGame.GetScreen()
renderer = Renderer.Create( Vector4.Create( 0, 0 ), Screen.GetSize( screen ) )
Renderer.SetTarget( renderer, screen )
```

See Also

- Renderer.Create
- Renderer.SetTarget
- Screen.GetSize
- Vector4.Create

ArcadeGame.LeaveGame

Brief

Leave an Arcade Game.

Definition

```
ArcadeGame.LeaveGame()
```

Arguments

None

Return Values

None.

Description

Tells the local player to leave the Arcade Game (if one is being played).

Examples

```
if (wantQuit) then
    ArcadeGame.LeaveGame()
end
```

See Also

None

AsyncCommand

AsyncCommand.Cancel

Brief

Cancels an in progress command.

Definition

```
boolean AsyncCommand.Cancel( AsyncCommand asyncCmd )
```

Arguments

asyncCmd - async command to cancel.

Return Values

It returns true if the command was successfully canceled and false otherwise.

Description

Cancels an executing command.

Examples

```
local result = AsyncCommand.Cancel( asyncCmd )
```

See Also

- AsyncCommand.Create
- AsyncCommand.Execute

AsyncCommand.Create

Brief

Create a new asynchronous command.

Definition

```
AsyncCommand.Create(commandType, ...)
```

Arguments

commandType - a valid command type value. See specific AsyncCommand.Create functions for valid command types.

Return Values

AsyncCommand object on success, nil on failure.

Description

Creates a new asynchronous command with the specified parameters, if any. If nil is returned and there is no associated Lua error provided, then this indicates that the specified command type is unavailable. If the command type is unavailable, the script should make no further attempts to create it.

Examples

```
ticketCmd = AsyncCommand.Create( AsyncCommands.TicketRequest )
```

See Also

- AsyncCommand.Create

- AsyncCommand.IsFinished
- AsyncCommand.GetErrorCode

AsyncCommand.Execute

Brief

Marks an asynchronous command for execution.

Definition

```
boolean AsyncCommand.Execute( asyncCmd, ...)
boolean AsyncCommand.Execute( AsyncCommand command, string serviceId, MemoryContainer container )
```

Arguments

asyncCmd - command to executeserviceId - The service id to request a ticket forcontainer - The container to store the ticket in

Return Values

false if command could not be executed, true if command successfully marked for execution.

Description

Submits the command to be executed. Note that success does not indicate that the command has finished, nor that it must successfully complete, rather that the command is in progress and will at some later stage either return successfully or with an error. The user should poll the command for completion, either within a coroutine or the game update loop. For the period between executing the command and receiving the result, the command is considered in progress. In this period the user can request cancellation of the command; this may or may not succeed depending on the execution stage of the command, and so should not be relied upon explicitly. No upper time limit is provided for the completion of command execution, nor are are any guarantees provided that commands will execute in the order they were submitted; in other words, a script should not rely on either a specific execution time or order. If required, execution order can be implemented by the script by waiting for preceding commands to finish before issuing dependent commands.

If Execute returns false, then this is most likely due to invalid or missing arguments passed to the function, and will be indicated by a Lua error - check the documentation for the command. If there is no Lua error, then the command execution has failed due to the NP services system being currently too busy to accept another command. In this case, wait some time and try to execute the command again, and repeat if necessary. This situation should be rare and transient, and only occur under very high load.

Examples

```
local container = MemoryContainer.Create( 1024 )
local ticketCmd = AsyncCommand.Create( AsyncCommands.TicketRequest )
AsyncCommand:Execute( ticketCmd, "XX1234-XXXX12345_00", container )
```

See Also

- AsyncCommand.Create
- AsyncCommand.IsFinished
- AsyncCommand.IsInProgress
- AsyncCommand.Cancel

AsyncCommand.GetErrorCode

Brief

Indicates the command result code if the command failed.

Definition

```
number AsyncCommand.GetErrorCode(asyncCmd)
```

Arguments

asyncCmd - async command to query.

Return Values

A result code indicating the error that occurred.
The result returned can be one of the following:
AsyncErr.Unspecified = 1
AsyncErr.System = 2
AsyncErr.Fatal = 3
AsyncErr.ServerError = 4
AsyncErr.SystemUnavailable = 5
AsyncErr.Aborted = 6
AsyncErr.PermissionDenied = 7
AsyncErr.LimitExceeded = 8
AsyncErr.AuthenticationFail = 10
AsyncErr.NotFound = 11
AsyncErr.Blacklisted = 12
AsyncErr.AlreadyExists = 13
AsyncErr.Timeout = 14
AsyncErr.Busy = 15
AsyncErr.Censored = 16

AsyncErrRanking.ScoreNotFound = 101
AsyncErrRanking.BoardNotFound = 102
AsyncErrRanking.InvalidScore = 103
AsyncErrRanking.NotBestScore = 105

AsyncErrTrophy.UnknownTitle = 300
AsyncErrTrophy.InvalidCommId = 301

Description

The result code is only valid if the command failed; the code should not expect any specific result code for a pending or successful command.
Result codes will be fixed, in that the same error will always produce the same result code.

Examples

```
if (not asyncCmd:Succeeded()) then
    print('Async command failed with result: ' .. asyncCmd:GetErrorCode())
end
```

See Also

- AsyncCommand.Succeeded

AsyncCommand.GetNpErrorCode

Brief

Gets the internal result code in the event of a command error result.

Definition

```
string AsyncCommand.GetNpErrorCode(asyncCmd)
```

Arguments

asyncCmd - async command to query.

Return Values

A result code indicating the error that occurred.

Description

The result code is only valid if the command failed; the code should not expect any specific result code for a pending or successful command. Result codes will be fixed, in that the same error will always produce the same result code. This command can be used to identify the cause of an error more exactly.

Examples

```
if (not asyncCmd:Succeeded()) then
    print('DEBUG: Async command failed with result: ' .. asyncCmd:GetNpErrorCode())
end
```

See Also

- AsyncCommand.Succeeded
- AsyncCommand.GetErrorCode

AsyncCommand.GetResultCount

Brief

Indicates the number of results available.

Definition

```
number AsyncCommand.GetResultCount( AsyncCommand asyncCmd )
```

Arguments

asyncCmd - async command to query.

Return Values

The number of results available.

Description

Returns the number of results available from the most recent execution of the command.

Examples

```
print(AsyncCommand.GetResultCount( asyncCmd ) .. ' results.')
```

See Also

AsyncCommand.GetResults

Brief

Access the results of the command.

Definition

```
value|table|array AsyncCommand.GetResults(AsyncCommand asyncCmd)
```

Arguments

asyncCmd - async command to query.

Return Values

A command-dependent value or table of values representing the results of the command.

Description

If the command has not successfully completed, nil will be returned. The exact format of the results is command-dependent, see the documentation for individual commands.

Examples

```
if (asyncCmd:IsFinished() and asyncCmd:Succeeded()) then
    print('Command results were: ', asyncCmd:GetResults())
end
```

See Also

- AsyncCommand.Succeeded

AsyncCommand.IsFinished

Brief

Indicates if the command has completed, either successfully or with an error.

Definition

```
boolean AsyncCommand.IsFinished(asyncCmd)
```

Arguments

asyncCmd - async command to query.

Return Values

true if the command has completed, false if not submitted or still in progress.

Description

Note that if a command has not yet been executed (only created), both IsInProgress and IsFinished will return false. This function will still return true if the command finishes with an error; to determine whether the command succeeded or failed query AsyncCommand.Succeeded().

Examples

```
if (asyncCmd:IsFinished() and asyncCmd:Succeeded()) then
    print('Results were: ', asyncCmd:GetResult())
end
```

See Also

- AsyncCommand.IsInProgress

- AsyncCommand.Succeeded
- AsyncCommand.GetErrorCode

AsyncCommand.IsInProgress

Brief

Indicates if the command has been submitted but has not yet completed.

Definition

```
boolean AsyncCommand.IsInProgress(asyncCmd)
```

Arguments

asyncCmd - async command to query.

Return Values

Returns true if command has been submitted and is not yet complete, or false otherwise.

Description

Note that if a command has not yet been executed (only created), both IsInProgress and IsFinished will return false.

Examples

```
if (asyncCmd:IsInProgress()) then
    print('Processing...')
end
```

See Also

- AsyncCommand.IsFinished

AsyncCommand.Succeeded

Brief

Indicates if the command succeeded or failed.

Definition

```
boolean AsyncCommand.Succeeded(asyncCmd)
```

Arguments

asyncCmd - async command to query.

Return Values

Returns true if the command is complete and succeeded, false otherwise.

Description

If the command has not been submitted or is still in progress, false will be returned. This should not be confused with the command failing, which can only be determined once the command is finished.

Examples

```
if (not asyncCmd:Succeeded()) then
    print('Async command failed with result: ', asyncCmd:GetErrorCode())
end
```

See Also

- AsyncCommand.GetErrorCode
- AsyncCommand.GetResults

AsyncContext

AsyncContext.Create

Brief

Creates a new asynchronous execution context.

Definition

```
AsyncContext AsyncContext.Create(AsyncContexts type, ...)
```

Arguments

type - the type of context to create. Can be one of the following:
AsyncContexts.TitleUserStorage
AsyncContexts.Trophy
AsyncContexts.ScoreRanking
AsyncContexts.Inventory
AsyncContexts.ObjectCatalog
AsyncContexts.Stream
AsyncContexts.SaveData

Return Values

The new async execution context, or nil if creation failed.

Description

Most async commands require the creation of an execution context in order to run. The context will typically specify details of the title and/or user the script wishes to target with the command. Note: It is good practice to nil your context if you will not use it again for the life of the script.

Examples

```
-- this creates a context to access the current user's TUS data for (non-existent) title NPWR99999
local passphrase = Resource.Find('passphrase')
local context = AsyncContext.Create(AsyncContexts.TitleUserStorage, "NPWR99999", passphrase)
```

See Also

- AsyncCommand.Create

Atmos

Atmos.GetParam

Brief

Reads a parameter from the scene atmosphere.

Definition

```
boolean|number|string|Vector4 Atmos.GetParam( string paramName )
boolean|number|string|Vector4 Atmos.GetParam( string paramName, number index )
```

Arguments

paramName - name of parameter to read.index - index of parameter. (if required)

Return Values

nil if parameter does not exist in scene. If successful the value of the parameter is returned.

Description

This function tries to retrieve a parameter value from the scene atmosphere, e.g. BASICSKY_HORIZONCOLOUR. If the parameter does not exist in the current atmosphere, nil is returned. The type of values returned back depend on the type of parameter. All parameter names and types can be found in the SkyTool documentation. Some parameters require an index because more than one can exist in a scene, e.g. SPRITECLOUD_TINTCOLOUR.

Examples

```
horizonColor = Atmos.GetParam( "BASICSKY_HORIZONCOLOUR" )           --  horizonColor will become a
Vector4
numSpriteClouds = Atmos.GetParam( "NUM_SPRITECLOUDS" )              --  numSpriteClouds will become a
number
spriteCloudTint = Atmos.GetParam( "SPRITECLOUD_TINTCOLOUR", 4 )     --  spriteCloudTint will become a
Vector4
```

See Also

None

Atmos.SetParam

Brief

Sets a parameter to the scene atmosphere

Definition

```
boolean Atmos.SetParam( string paramName, boolean|number|string|Vector4 paramVal )
boolean Atmos.SetParam( string paramName, number index, boolean|number|string|Vector4 paramVal )
```

Arguments

paramName - name of parameter to be set.index - index of parameter. (if required)paramVal - value of parameter to be set.

Return Values

false if parameter does not exist in scene, or if the type given is incorrect. true if parameter set correctly.

Description

This function sets a parameter in the scene atmosphere, e.g. BASICSKY_HORIZONCOLOUR. If the parameter does not exist in the current atmosphere, false is returned. The type of value to assign depends on the type of parameter. If the type given is incorrect, false is returned. All parameter names and types can be found in the SkyTool documentation. Access to the Atmos library is only granted for scene objects such as mini games and embedded objects. Some parameters require an index because more than one can exist in a scene, e.g. SPRITECLOUD_TINTCOLOUR.

Examples

```
Atmos.SetParam( "BASICSKY_HORIZONCOLOUR", horizonColor );          --  horizonColor is a Vector4
Atmos.SetParam( "EXPHEIGHTFOG_FOGDENSITY", fogDensity );           --  fogDensity is a number
Atmos.SetParam( "SPRITECLOUD_TINTCOLOUR", 4, spriteCloudTint );    --  spriteCloudTint is a Vector4
```

See Also

None

Base64

Base64.Decode

Brief

Decodes the base64 contents of a memory container.

Definition

```
number Base64.Decode( MemoryContainer out, MemoryContainer in, number offset = -1, number size = -1 )
```

Arguments

out - The memory container to write the decoded data to.in - The memory container to read the base64 data from.offset - The offset of the data in the input memory container. Defaults to -1.size - The size of data to read from the input memory container. Defaults to -1.

Return Values

The number of bytes written to the output memory container or a negative number on error.
-1 - Invalid arg
-2 - Not enough space in the target memory container

Description

Decodes the base64 contents of a memory container. Data is always written to the output container as a stream. If no size or offset are supplied (or are set to -1), all available data from the input memory container will be encoded. This function will not alter the input memory container in anyway.

Examples

```
local output = MemoryContainer.Create( 1024 )
local numBytesDecoded = Base64.Decode( output, input )
```

See Also

- Base64.Encode
- Base64.DecodeFromString

Base64.DecodeFromString

Brief

Decodes the base64 contents of a string.

Definition

```
number Base64.DecodeFromString( MemoryContainer out, string in )
```

Arguments

out - The memory container to write the decoded data to.in - The string containing the base64 data.

Return Values

The number of bytes written to the output memory container or a negative number on error.
-1 - Invalid arg
-2 - Not enough space in the target memory container

Description

Decodes the base64 string and writes the decoded data to the output memory container. Data is always written to the output container as a stream.

Examples

```
local output = MemoryContainer.Create( 1024 )
local numBytesDecoded = Base64.DecodeFromString( output, 'QmFzZTY0IHRlc3QgMw==' )
print( MemoryContainer.GetString( output, 0 ) )
```

See Also

- Base64.Encode
- Base64.Decode

Base64.Encode

Brief

Encodes the contents of a memory container into base64.

Definition

```
number Base64.Encode( MemoryContainer out, MemoryContainer in, number offset = -1, number size = -1 )
```

Arguments

out - The memory container to write the encoded data to.in - The memory container to read the original data from.offset - The offset of the data in the input memory container.size - The size of data to read from the input memory container.

Return Values

The number of bytes written to the output memory container or a negative number on error.
-1 - Invalid arg
-2 - Not enough space in the target memory container

Description

Encodes the contents of a memory container into base64. Data is always written to the output container as a stream. If no size or offset are supplied (or are set to -1), all available data from the input memory container will be encoded. This function will not alter the input memory

container in anyway.

Examples

```
local input = MemoryContainer.Create( 1024 )
local output = MemoryContainer.Create( 1024 )
MemoryContainer.SetString( input, 0, 'Base64 test 1' )
local numBytesEncoded = Base64.Encode( output, input )
```

See Also

- Base64.Decode
- Base64.DecodeFromString

BasicGenx

BasicGenx.AddAttribute

Brief

Adds an attribute to the current element.

Definition

```
number BasicGenx.AddAttribute( BasicGenx writer, string name, string|boolean|number value )
```

Arguments

writer - The instance of the XML writer.name - The name of the new attribute.value - A string, boolean or number to be used as the attribute value.

Return Values

0 on success, any other value is an error. BasicGenx error codes are:
1 - Bad UTF-8
2 - Non-XML character
3 - Bad name
4 - Memory allocation failed
5 - Bad namespace name
6 - Internal error
7 - Duplicate prefix
8 - Call out of sequence
9 - No start tag for EndElement call
10 - I/O error
11 - Missing attribute value
12 - Malformed comment body
13 - ?> in Processing Instruction
14 - Target of Processing Instruction matches [xX][mM][lL]
15 - Same attribute specified more than once
16 - Attribute cannot be in default namespace
17 - Namespace declared twice with different prefixes on one element
18 - Default namespace declared on an element which is in no namespace

Description

Adds an attribute to the current element. If the value passed is a boolean and it is set to true, then "true" is written, otherwise, "false" is written. If the value passed is a number, it will always be written in floating point representation even if it is a whole number. In order to write a number in integer format, it is necessary to use BasicGenx.AddAttributeInt. A string value is written as specified. Passing nil as the value parameter will result in the string "nil" being written.

Examples

```
-- Example 1: Write "Hello World" to the XML
stringValue = "John Smith"
result = BasicGenx.AddAttribute( writer, "playerName", stringValue )

-- Example 2: Write "true" to the XML
booleanValue = true
result = BasicGenx.AddAttribute( writer, "enableStats", booleanValue )

-- Example 3: Write "100.000000" to the XML
numberValue = 100
result = BasicGenx.AddAttribute( writer, "percentComplete", numberValue )
```

See Also

- BasicGenx.AddAttributeInt

BasicGenx.AddAttributeInt

Brief

Adds a new attribute to the current XML element with integer formatting.

Definition

```
number BasicGenx.AddAttributeInt( BasicGenx writer, string name, number value )
```

Arguments

writer - The instance of the XML writer.name - The name of the new attribute.value - The value of the attribute.

Return Values

0 on success, any other value is an error. BasicGenx error codes are:
1 - Bad UTF-8
2 - Non-XML character
3 - Bad name
4 - Memory allocation failed
5 - Bad namespace name
6 - Internal error
7 - Duplicate prefix
8 - Call out of sequence
9 - No start tag for EndElement call
10 - I/O error
11 - Missing attribute value
12 - Malformed comment body
13 - ?> in Processing Instruction
14 - Target of Processing Instruction matches [xX][mM][lL]
15 - Same attribute specified more than once
16 - Attribute cannot be in default namespace
17 - Namespace declared twice with different prefixes on one element
18 - Default namespace declared on an element which is in no namespace

Description

Adds a new attribute to the current XML element with integer formatting. The value is written specifically as an integer, as opposed to BasicGenx.AddAttribute which writes numbers in floating point format. If the specified number is not an integer, it is truncated in some non-specified way.

Examples

```
local result = BasicGenx.AddAttributeInt( writer, "numItems", 10 )
```

See Also

- BasicGenx.AddAttribute

BasicGenx.AddText

Brief

Sets the value of an XML element.

Definition

```
number BasicGenx.AddText( BasicGenx writer, string|boolean|number value )
```

Arguments

writer - The instance of the XML writer.value - A string, boolean or number to set as the value in the XML element.

Return Values

0 on success, any other value is an error. BasicGenx error codes are:
1 - Bad UTF-8
2 - Non-XML character
3 - Bad name
4 - Memory allocation failed
5 - Bad namespace name
6 - Internal error
7 - Duplicate prefix
8 - Call out of sequence
9 - No start tag for EndElement call
10 - I/O error
11 - Missing attribute value
12 - Malformed comment body
13 - ?> in Processing Instruction
14 - Target of Processing Instruction matches [xX][mM][lL]
15 - Same attribute specified more than once
16 - Attribute cannot be in default namespace
17 - Namespace declared twice with different prefixes on one element
18 - Default namespace declared on an element which is in no namespace

Description

Sets the value of an XML element. If the value passed is a boolean and it is set to true, then "true" is written, otherwise, "false" is written. If the value passed is a number, it will always be written in floating point representation even if it is a whole number. In order to write a number in integer format, it is necessary to use BasicGenx.AddTextAsInt. A string value is written as specified. Passing nil as the value parameter will result in the string "nil" being written.

Examples

```
-- Example 1: Write "Hello World" to the XML
stringValue = "Hello World"
result = BasicGenx.AddText( writer, stringValue )

-- Example 2: Write "true" to the XML
booleanValue = true
result = BasicGenx.AddText( writer, booleanValue )

-- Example 3: Write "10.000000" to the XML
numberValue = 10
result = BasicGenx.AddText( writer, numberValue )
```

See Also

- BasicGenx.AddTextAsInt

BasicGenx.AddTextAsInt

Brief

Sets the value of an XML element with integer formatting.

Definition

```
number BasicGenx.AddTextAsInt( BasicGenx writer, number value )
```

Arguments

writer - The instance of the XML writer.value - The value to set in the XML element.

Return Values

0 on success, any other value is an error. BasicGenx error codes are:
1 - Bad UTF-8
2 - Non-XML character
3 - Bad name
4 - Memory allocation failed
5 - Bad namespace name
6 - Internal error
7 - Duplicate prefix
8 - Call out of sequence
9 - No start tag for EndElement call
10 - I/O error
11 - Missing attribute value
12 - Malformed comment body
13 - ?> in Processing Instruction
14 - Target of Processing Instruction matches [xX][mM][lL]
15 - Same attribute specified more than once
16 - Attribute cannot be in default namespace
17 - Namespace declared twice with different prefixes on one element
18 - Default namespace declared on an element which is in no namespace

Description

Sets the value of an XML element with integer formatting. The value is written specifically as an integer, as opposed to BasicGenx.AddText which writes numbers in floating point format. If the specified number is not an integer, it is truncated in some non-specified way.

Examples

```
-- Example: Write "10" to the XML
numberValue = 10
result = BasicGenx.AddTextAsInt( writer, numberValue )
```

See Also

- BasicGenx.AddText

BasicGenx.CheckText

Brief

Scan a string and report problems.

Definition

```
number BasicGenx.CheckText( BasicGenx writer, string text )
```

Arguments

writer - The instance of the XML writer.text - The string to check.

Return Values

0 on success, any other value is an error. BasicGenx error codes are:
1 - Bad UTF-8
2 - Non-XML character
3 - Bad name
4 - Memory allocation failed
5 - Bad namespace name
6 - Internal error
7 - Duplicate prefix
8 - Call out of sequence
9 - No start tag for EndElement call
10 - I/O error
11 - Missing attribute value
12 - Malformed comment body
13 - ?> in Processing Instruction
14 - Target of Processing Instruction matches [xX][mM][lL]
15 - Same attribute specified more than once
16 - Attribute cannot be in default namespace
17 - Namespace declared twice with different prefixes on one element
18 - Default namespace declared on an element which is in no namespace

Description

Scan a string to be written by BasicGenx and report problems with UTF-8 encoding or non-XML characters.

Examples

```
local result = BasicGenx.CheckText( writer, "My test text" )
```

See Also

None

BasicGenx.Create

Brief

Creates a new XML writer.

Definition

```
BasicGenx.Create( MemoryContainer container )
```

Arguments

container - The memory container to write the XML data to.

Return Values

A new instance of an XML writer.

Description

Creates a new XML writer. The memory container is written to as if it were a stream so it can be used to write data to an existing container.

Examples

```
local container = MemoryContainer.Create( 4096 )
local writer = BasicGenx.Create( container )
```

See Also

- MemoryContainer.Create

BasicGenx.EndDocument

Brief

Ends writing an XML document.

Definition

```
BasicGenx.EndDocument( BasicGenx writer )
```

Arguments

writer - The instance of the XML writer.

Return Values

None.

Description

Ends writing an XML document.

Examples

```
local container = MemoryContainer.Create( 4096 )
local writer = BasicGenx.Create( container )
BasicGenx.StartDocument( writer )

-- Write XML element hierarchy here...

BasicGenx.EndDocument( writer )
```

See Also

- BasicGenx.StartDocument

BasicGenx.EndElement

Brief

Closes an XML element.

Definition

```
number BasicGenx.EndElement( BasicGenx writer )
```

Arguments

writer - The instance of the XML writer.

Return Values

0 on success, any other value is an error. BasicGenx error codes are:
1 - Bad UTF-8
2 - Non-XML character
3 - Bad name
4 - Memory allocation failed
5 - Bad namespace name
6 - Internal error
7 - Duplicate prefix
8 - Call out of sequence
9 - No start tag for EndElement call
10 - I/O error
11 - Missing attribute value
12 - Malformed comment body
13 - ?> in Processing Instruction
14 - Target of Processing Instruction matches [xX][mM][lL]
15 - Same attribute specified more than once
16 - Attribute cannot be in default namespace
17 - Namespace declared twice with different prefixes on one element
18 - Default namespace declared on an element which is in no namespace

Description

Closes an XML element which must have previously been opened with BasicGenx.StartElement.

Examples

```
local result = BasicGenx.EndElement( writer )
```

See Also

- BasicGenx.StartElement

BasicGenx.StartDocument

Brief

Starts writing an XML document.

Definition

```
BasicGenx.StartDocument( BasicGenx writer )
```

Arguments

writer - The instance of the XML writer.

Return Values

None.

Description

Starts writing an XML document.

Examples

```
local container = MemoryContainer.Create( 4096 )
local writer = BasicGenx.Create( container )
BasicGenx.StartDocument( writer )

-- Write XML element hierarchy here...

BasicGenx.EndDocument( writer )
```

See Also

- BasicGenx.Create
- BasicGenx.EndDocument
- MemoryContainer.Create

BasicGenx.StartElement

Brief

Starts writing an XML element.

Definition

```
number BasicGenx.StartElement( BasicGenx writer, string name )
```

Arguments

writer - The instance of the XML writer.name - The name of the new XML element.

Return Values

0 on success, any other value is an error. BasicGenx error codes are:
1 - Bad UTF-8
2 - Non-XML character
3 - Bad name
4 - Memory allocation failed
5 - Bad namespace name
6 - Internal error
7 - Duplicate prefix
8 - Call out of sequence
9 - No start tag for EndElement call
10 - I/O error
11 - Missing attribute value
12 - Malformed comment body
13 - ?> in Processing Instruction
14 - Target of Processing Instruction matches [xX][mM][lL]
15 - Same attribute specified more than once
16 - Attribute cannot be in default namespace
17 - Namespace declared twice with different prefixes on one element
18 - Default namespace declared on an element which is in no namespace

Description

Starts writing an XML element. It is valid to nest calls to BasicGenx.StartElement.

Examples

```
local result = BasicGenx.StartElement( writer, "XML" )
```

See Also

- BasicGenx.EndElement

BigInt

BigInt.Add

Brief

Add the specified value to a BigInt.

Definition

```
BigInt.Add(BigInt dst, BigInt|number|string src1, BigInt|number|string src2)
```

Arguments

dst - the BigInt to modifysrc1 - the first/left sourcesrc2 - the second/right source

Return Values

Description

Adds the specified values to the specified BigInt. dst = src1 + src2.

Examples

```
bigInt1:Add(bigInt1, 32)
bigInt2:Add(bigInt1, "0x393823")
bigInt3:Add("723523", bigInt3)
bigInt4:Add(bigInt3, bigInt3)
```

See Also

- BigInt.Create

BigInt.And

Brief

Perform a bitwise AND operation with a BigInt and a specified value.

Definition

```
BigInt.And(BigInt dst, BigInt|number|string src1, BigInt|number|string src2)
```

Arguments

dst - the BigInt to modifysrc1 - the first/left sourcesrc2 - the second/right source

Return Values

Description

Perform a bitwise AND operation with a BigInt and a specified value. String values may be decimal, or if prefixed with 0x or 0X, hexadecimal. The string will be checked to ensure it is a valid number and fits within the BigInt's precision limit.

Examples

```
bigInt1:And(32)
bigInt2:And("0x8000")
bigInt3:And("128")
bigInt4:And(bigInt3)
```

See Also

- BigInt.Create

BigInt.AsHexString

Brief

Generates a hexadecimal string representation of the BigInt.

Definition

```
string BigInt.AsHexString(BigInt i)
```

Arguments

i - a BigInt

Return Values

A string containing the BigInt value in hexadecimal.

Description

Generates a hexadecimal string representation of the BigInt. The string will contain only hexadecimal digits (0-9, A-F) with no prefix (0x or similar).

Examples

```
local i = BigInt.Create(64, "ABCDEF01234456789")
print(i:AsHexString())      -- prints "ABCDEF01234456789"
```

See Also

- BigInt.Create

BigInt.Create

Brief

Create a signed big integer with the specified number of bits.

Definition

```
BigInt BigInt.Create(number numberOfBits, number|BigInt|string initialValue = 0 )
```

Arguments

numberOfBits - size of integer in bits
initialValue - value to initialise returned big integer in either number, string or BigInt.

Return Values

A big integer with the specified size, or nil on failure.

Description

Create a big integer with the specified number of bits of precision. At present only 32 and 64 bit integers are supported. Note that these are not "native" types and therefore have a significant performance penalty, and should only be used where necessary. Normal integer arithmetic should be done using the built-in Lua number type. String values may be either decimal or, if prefixed with 0x or 0X, hexadecimal. The string will be checked to ensure it is a valid number and fits within the BigInt's precision limit.

Examples

```
local bigInt = BigInt.Create(64, "0x1234567890ABCDEF")
local bigInt2 = BigInt.Create(64, "1234567890123456")
local bigInt3 = BigInt.Create(32, 123321)
local bigInt4 = BigInt.Create(32, bigInt3)
```

See Also

- BigInt.Set

BigInt.Divide

Brief

Divide a BigInt by a specified value.

Definition

```
BigInt.Divide(BigInt dst, BigInt|number|string src1, BigInt|number|string src2)
```

Arguments

dst - the BigInt to modifysrc1 - the first/left sourcesrc2 - the second/right source

Return Values

Description

Divide the specified BigInt by the specified value. String values may be decimal, or if prefixed with 0x or 0X, hexadecimal. The string will be checked to ensure it is a valid number and fits within the BigInt's precision limit.

Examples

```
bigInt1:Divide(32)
bigInt2:Divide("0x393823")
bigInt3:Divide("723523")
bigInt4:Divide(bigInt3)
```

See Also

- BigInt.Create

BigInt.GetBit

Brief

Returns the value of the specified bit.

Definition

```
boolean BigInt.GetBit( BigInt i, number bitIndex )
```

Arguments

i - a BigIntbitIndex - Which bit to get the value of. bitIndex starts at 0.

Return Values

Returns the value of the specified bit.

Description

Returns the value of the specified bit.

Examples

```
local i = BigInt.Create( 64, -1 )
print( 'top bit is ', BigInt.GetBit(i, 63 ) )
```

See Also

- BigInt.Create
- BigInt.SetBit

BigInt.Multiply

Brief

Multiply the specified value with a BigInt.

Definition

```
BigInt.Multiply(BigInt dst, BigInt|number|string src1, BigInt|number|string src2)
```

Arguments

dst - the BigInt to modifysrc1 - the first/left sourcesrc2 - the second/right source

Return Values

Description

Multiply the specified value with the specified BigInt. String values may be decimal, or if prefixed with 0x or 0X, hexadecimal. The string will be checked to ensure it is a valid number and fits within the BigInt's precision limit.

Examples

```
bigInt1:Multiply(32)
bigInt2:Multiply("0x393823")
bigInt3:Multiply("723523")
bigInt4:Multiply(bigInt3)
```

See Also

- BigInt.Create

BigInt.Negate

Brief

Negate a BigInt.

Definition

```
BigInt.Negate(BigInt i)
```

Arguments

i - the BigInt to negate

Return Values

Description

Negate the specified BigInt, that is, if it's current value is x, set it to -x.

Examples

```
bigInt1:Negate()
```

See Also

- BigInt.Create

BigInt.Not

Brief

Perform a bitwise NOT operation on the specified BigInt.

Definition

```
BigInt.Not(BigInt i)
```

Arguments

i - the BigInt on which to perform bitwise NOT

Return Values

Description

Perform a bitwise NOT operation on the specified BigInt.

Examples

```
bigInt1:Not()
```

See Also

- BigInt.Create

BigInt.Or

Brief

Perform a bitwise OR operation with a BigInt and a specified value.

Definition

```
BigInt.Or(BigInt dst, BigInt|number|string src1, BigInt|number|string src2)
```

Arguments

dst - the BigInt to modifysrc1 - the first/left sourcesrc2 - the second/right source

Return Values

Description

Perform a bitwise OR operation with a BigInt and a specified value. String values may be decimal, or if prefixed with 0x or 0X, hexadecimal. The string will be checked to ensure it is a valid number and fits within the BigInt's precision limit.

Examples

```
bigInt1:Or(32)
bigInt2:Or("0x8000")
bigInt3:Or("128")
bigInt4:Or(bigInt3)
```

See Also

- BigInt.Create

BigInt.Set

Brief

Set a big integer to the specified value

Definition

```
BigInt.Set(BigInt|number|string value)
```

Arguments

value - value to set

Return Values

Description

Set a big integer to the specified value. String values may be either decimal or, if prefixed with 0x or 0X, hexadecimal. The string will be checked to ensure it is a valid number and fits within the BigInt's precision limit.

Examples

```
bigInt:Set("0x1234567890ABCDEF")
bigInt2:Set("1234567890123456")
bigInt3:Set(123321)
bigInt4:Set(bigInt3)
```

See Also

- BigInt.Create

BigInt.SetBit

Brief

Sets the the specified bit to the specified value.

Definition

```
boolean BigInt.SetBit( BigInt i, number bitIndex, boolean value )
```

Arguments

i - a BigIntbitIndex - Which bit to get the value of. bitIndex starts at 0.value - The value to set the bit to, true is on and false is off.

Return Values

None

Description

Sets the the specified bit to the specified value.

Examples

```
local i = BigInt.Create( 64, 1234 )
BigInt.SetBit( i, 63, true ) ) -- Make it negative.
```

See Also

- BigInt.Create
- BigInt.GetBit

BigInt.SetZero

Brief

Set the specified BigInt to zero.

Definition

```
BigInt.SetZero(BigInt i)
```

Arguments

i - the BigInt to set to zero

Return Values

Description

Set the specified BigInt to zero.

Examples

```
bigInt1:SetZero()
```

See Also

- BigInt.Create

BigInt.ShiftLeft

Brief

Bit shifts the source integer left by the specified amount of bits and saves in the destination.

Definition

```
BigInt.ShiftLeft( BigInt dst, BigInt|number|string src, BigInt|number|string shift )
```

Arguments

dst - The BigInt to save the result to.src - The value to shift.shift - the amount to shift by.

Return Values

None

Description

Bit shifts the source integer left by the specified amount of bits and saves the result in the destination.

Examples

```
dst = BigInt.Create( 64 )
src = BigInt.Create( 64, "0xFF0000" )
BigInt.ShiftLeft( dst, src, 5 )
```

See Also

- BigInt.Create
- BigInt.ShiftRight
- BigInt.GetBit
- BigInt.SetBit

BigInt.ShiftRight

Brief

Bit shifts the source integer right by the specified amount of bits and saves the result in the destination.

Definition

```
BigInt.ShiftRight( BigInt dst, BigInt|number|string src, BigInt|number|string shift )
```

Arguments

dst - The BigInt to save the result to.src - The value to shift.shift - the amount to shift by.

Return Values

None

Description

Bit shifts the source integer right by the specified amount of bits and saves the result in the destination.

Examples

```
dst = BigInt.Create( 64 )
src = BigInt.Create( 64, "0xFF0000" )
BigInt.ShiftRight( dst, src, 5 )
```

See Also

- BigInt.Create
- BigInt.ShiftRight
- BigInt.GetBit
- BigInt.SetBit

BigInt.Subtract

Brief

Subtract the specified value from a BigInt.

Definition

```
BigInt.Subtract(BigInt dst, BigInt|number|string src1, BigInt|number|string src2)
```

Arguments

dst - the BigInt to modifysrc1 - the first/left sourcesrc2 - the second/right source

Return Values

Description

Subtract the specified value from the specified BigInt. String values may be decimal, or if prefixed with 0x or 0X, hexadecimal. The string will be checked to ensure it is a valid number and fits within the BigInt's precision limit.

Examples

```
bigInt1:Subtract(32)
bigInt2:Subtract("0x393823")
bigInt3:Subtract("723523")
bigInt4:Subtract(bigInt3)
```

See Also

- BigInt.Create

BigInt.ToNumber

Brief

Converts a BigInt to a native Lua number value if possible.

Definition

```
number BigInt.ToNumber( BigInt i )
```

Arguments

i - a BigInt

Return Values

The Lua number value equivalent to the BigInt.

Description

Converts a BigInt to a native Lua number value if possible. If the BigInt is out of range, a Lua error will occur.

Examples

```
local i = BigInt.Create( 32, 1234 )
print(i:ToNumber() + 1)     -- prints 1235
i:Set("121223523262")
--print(i:ToNumber())       this will error
```

See Also

- BigInt.Create

BigInt.Xor

Brief

Perform a bitwise XOR operation with a BigInt and a specified value.

Definition

```
BigInt.Xor(BigInt dst, BigInt|number|string src1, BigInt|number|string src2)
```

Arguments

dst - the BigInt to modifysrc1 - the first/left sourcesrc2 - the second/right source

Return Values

Description

Perform a bitwise XOR operation with a BigInt and a specified value. String values may be decimal, or if prefixed with 0x or 0X, hexadecimal. The string will be checked to ensure it is a valid number and fits within the BigInt's precision limit.

Examples

```
bigInt1:Xor(32)
bigInt2:Xor("0x8000")
bigInt3:Xor("128")
bigInt4:Xor(bigInt3)
```

See Also

- BigInt.Create

Camera

Camera.Activate

Brief

Activates the specified camera.

Definition

```
Camera.Activate( Camera camera, number transitionTime = 0.0 )
```

Arguments

camera - The camera to activate.transitionTime - The time taken to smoothly move from the previously active camera's position, to the specified camera's position.

Return Values

None.

Description

Activates the specified camera.

Examples

```
Camera.Activate( gameCamera, 1.0 )
```

See Also

- Camera.Deactivate
- Camera.IsActivated

Camera.AttachToParent

Brief

Links the camera to the given entity so that the camera inherits the entity's transform.

Definition

```
Camera.AttachToParent( Camera camera, Entity parent )
```

Arguments

camera - The camera to update.parent - The entity to attach this camera to, or nil to detach the camera.

Return Values

None.

Description

Links the camera to the given entity so that the camera inherits the entity's transform. This will override any values set via Camera.SetPosition, or Camera.SetLookAtPosition until the connection is cleared by passing in nil as the parent. If the entity is destroyed then the camera will be unparented. It is not possible at present to specify a local transform offset from the entity.

Examples

```
Camera.AttachToParent( gameCamera, myEntity ) -- connect the camera to myEntity
Camera.AttachToParent( gameCamera, nil ) -- detach the camera from any connected entity
```

See Also

None

Camera.Create

Brief

Creates a new 3D camera.

Definition

```
Camera Camera.Create( string constructionMethod = nil )
```

Arguments

constructionMethod - How to construct the camera, either nil or "current".

Return Values

Returns a Camera instance.

Description

Creates a new 3D camera. This camera is, by default, positioned at the origin (0, 0, 0). If constructionMethod is "current" it creates a new camera with the properties of the currently active camera.

Examples

```
gameCamera = Camera.Create( "current" )
Camera.SetPosition( gameCamera, Camera.GetPosition() + Vector4.Create( 0, 1, 0 ) )
```

See Also

- Camera.Activate

Camera.Deactivate

Brief

Deactivates the specified camera.

Definition

```
Camera.Deactivate( Camera camera, number transitionTime = 0.0 )
```

Arguments

camera - The camera to deactivate.transitionTime - The time taken to smoothly move from the specified camera's position back to the previously active camera's position.

Return Values

None.

Description

Deactivates the specified camera.

Examples

```
if Camera.IsActivated( gameCamera ) then
    Camera.Deactivate( gameCamera, 1.0 )
end
```

See Also

- Camera.Activate
- Camera.IsActivated

Camera.EnableAvatarCameraSync

Brief

Tell the normal Avatar camera to face the same direction as a deactivating custom camera, minimizing the camera transition.

Definition

```
Camera.EnableAvatarCameraSync()
```

Arguments

None

Return Values

Description

When enabled, the normal Avatar Camera will sync with any active custom camera to face the same direction. This simply means that when the custom camera is deactivated, the normal Avatar Camera will be facing the same direction, and the magnitude of the transition between the two will be minimized. This defaults to on. It's rare, and generally not advised to disable this.

Examples

```
Camera.EnableAvatarCameraSync(false)
```

See Also

None

Camera.GetInverseViewMatrix

Brief

Get the view to world transform for the currently active camera.

Definition

```
Camera.GetInverseViewMatrix(Matrix44 matrix)
```

Arguments

matrix - The matrix to set.

Return Values

None.

Description

Retrieves the view to world transform for the currently active camera. This can be considered as the world transform of the camera object itself.

Examples

```
local invViewMat = Matrix44.Create()
Camera.GetInverseViewMatrix(invViewMat)
```

See Also

- Camera.GetViewMatrix
- Camera.GetProjectionMatrix

Camera.GetLookAtPosition

Brief

Retrieves the camera's world space target position.

Definition

```
Vector4 Camera.GetLookAtPosition( Camera camera )
Camera.GetLookAtPosition( Camera camera, Vector4 outPos )
Vector4 Camera.GetLookAtPosition()
Camera.GetLookAtPosition( Vector4 outPos )
```

Arguments

camera - The camera to query.outPos - (out) The vector to receive the target position.

Return Values

Returns a Vector4 specifying the world space target position of the camera.

Description

Retrieves the camera's world space target position. Only the X, Y and Z components of the Vector4 are valid. If no camera is specified it uses the currently active camera.

Examples

```
camtarget = Camera.GetLookAtPosition( gameCamera )
```

See Also

- Camera.SetLookAtPosition

Camera.GetPosition

Brief

Retrieves the camera's world space position.

Definition

```
Vector4 Camera.GetPosition( Camera camera )
Camera.GetPosition( Camera camera, Vector4 outPos )
Vector4 Camera.GetPosition()
Camera.GetPosition( Vector4 outPos )
```

Arguments

camera - The camera to query.outPos - (out) The vector to receive the camera position.

Return Values

A Vector4 specifying the world space position of the camera.

Description

Retrieves the camera's world space position. Only the X, Y and Z components of the Vector4 are valid. If no camera is specified it uses the currently active camera. Note that inactive cameras are not updated, and therefore if a position is requested for an inactive camera the most recent position available will be returned. For the currently active camera the position will match the translation of the inverse view matrix.

Examples

```
campos = Camera.GetPosition( gameCamera )
```

See Also

- Camera.SetPosition
- Camera.GetInverseViewMatrix

Camera.GetProjectionMatrix

Brief

Get the projection matrix for the currently active camera.

Definition

```
Camera.GetProjectionMatrix(Matrix44 matrix)
```

Arguments

matrix - The matrix to set.

Return Values

None.

Description

Retrieves the projection transform for the currently active camera. A script should not assume a particular projection transform format. The projection matrix is used to transform a point from view to clip space.

Examples

```
local projMat = Matrix44.Create()
Camera.GetProjectionMatrix(projMat)
```

See Also

- Camera.GetViewMatrix
- Camera.GetInverseViewMatrix

Camera.GetViewMatrix

Brief

Get the world to view transform for the currently active camera.

Definition

```
Camera.GetViewMatrix(Matrix44 matrix)
```

Arguments

matrix - The matrix to set.

Return Values

None.

Description

This matrix will transform from world space to view space.

Examples

```
local worldToViewMat = Matrix44.Create()
Camera.GetViewMatrix(worldToViewMat)
```

See Also

- Camera.GetInverseViewMatrix
- Camera.GetProjectionMatrix

Camera.GetViewPortFOV

Brief

Retrieves the vertical field of view for the specified camera.

Definition

```
number Camera.GetViewPortFOV( Camera camera )
number Camera.GetViewPortFOV()
```

Arguments

camera - The camera to query.

Return Values

The vertical field of view for the camera, specified in degrees.

Description

Retrieves the vertical field of view (FOV) for the specified camera. If no camera is specified it uses the currently active camera.

Examples

```
fov = Camera.GetViewPortFOV( gameCamera )
```

See Also

- Camera.SetViewPortFOV

Camera.GetViewPortZFar

Brief

Retrieves the camera's far clip plane.

Definition

```
number Camera.GetViewPortZFar( Camera camera )
number Camera.GetViewPortZFar()
```

Arguments

camera - The camera to query.

Return Values

The far clip plane of the camera in units of meters.

Description

Retrieves the camera's far clip plane. Units are specified in meters. The default far clip plane for a newly created camera is 4000.0. If no camera is specified it uses the currently active camera.

Examples

```
farz = Camera.GetViewPortZFar( gameCamera )
```

See Also

- Camera.SetViewPortZFar

Camera.GetViewPortZNear

Brief

Retrieves the camera's near clip plane.

Definition

```
number Camera.GetViewPortZNear( Camera camera )
number Camera.GetViewPortZNear()
```

Arguments

camera - The camera to query.

Return Values

The near clip plane of the camera in units of meters.

Description

Retrieves the camera's near clip plane. Units are specified in metres. The default near clip plane for a newly created camera is 0.1. If no camera is specified it uses the currently active camera.

Examples

```
nearz = Camera.GetViewPortZNear( gameCamera )
```

See Also

- Camera.SetViewPortZNear

Camera.IsActivated

Brief

Determines whether the specified camera is currently active.

Definition

```
boolean Camera.IsActivated( Camera camera )
```

Arguments

camera - The camera to query.

Return Values

true if the specified camera is activated, false otherwise.

Description

Determines whether the specified camera is currently active.

Examples

```
if Camera.IsActivated( gameCamera ) then
    Camera.Dectivate( gameCamera, 1.0 )
end
```

See Also

- Camera.Activate
- Camera.Deactivate

Camera.SetLookAtPosition

Brief

Sets the camera's world space target position.

Definition

```
Camera.SetLookAtPosition( Camera camera, Vector4 target )
```

Arguments

camera - The camera to update.target - Vector4 specifying the world space target position for the camera.

Return Values

None.

Description

Sets the camera's world space target position. Only the X, Y and Z components of the Vector4 are used to position the camera. The W component is ignored.

Examples

```
Camera.SetLookAtPosition( gameCamera, Vector4.Create( 0, 10, 0, 0 ) )
```

See Also

- Camera.GetLookAtPosition

Camera.SetPosition

Brief

Sets the camera's world space position.

Definition

```
Camera.SetPosition( Camera camera, Vector4 position )
```

Arguments

camera - The camera to update.position - Vector4 specifying the world space position of the camera.

Return Values

None.

Description

Sets the camera's world space position. Only the X, Y and Z components of the Vector4 are used to position the camera. The W component is ignored.

Examples

```
Camera.SetPosition( gameCamera, Vector4.Create( 0, 10, 0, 0 ) )
```

See Also

- Camera.GetPosition

Camera.SetViewPortAspect

Brief

Sets the aspect ratio for the specified camera's viewport.

Definition

```
Camera.SetViewPortAspect( Camera camera, number aspect )
```

Arguments

camera - The camera to update.aspect - aspect ratio for the camera's viewport (width / height).

Return Values

None.

Description

Sets the aspect ratio for the specified camera's viewport.

Examples

```
Camera.SetViewPortAspect( gameCamera, 16.0 / 9.0 )
```

See Also

Camera.SetViewPortFOV

Brief

Sets the vertical field of view of the specified camera.

Definition

```
Camera.SetViewPortFOV( Camera camera, number fov )
```

Arguments

camera - The camera to update.fov - Vertical field of view for the camera, specified in degrees.

Return Values

None.

Description

Sets the vertical field of view of the specified camera in degrees.

Examples

```
Camera.SetViewPortFOV( gameCamera, 60.0 )
```

See Also

- Camera.GetViewPortFOV

Camera.SetViewPortZFar

Brief

Sets the camera's far clip plane.

Definition

```
Camera.SetViewPortZFar( Camera camera, number zfar )
```

Arguments

camera - The camera to update.znear - The far clip plane of the camera in units of metres.

Return Values

None.

Description

Sets the camera's far clip plane. Units are specified in metres. The default far clip plane for a newly created camera is 4000.0.

Examples

```
Camera.SetViewPortZFar( gameCamera, 5000.0 )
```

See Also

- Camera.GetViewPortZFar

Camera.SetViewPortZNear

Brief

Sets the camera's near clip plane.

Definition

```
Camera.SetViewPortZNear( Camera camera, number znear )
```

Arguments

camera - The camera to update.znear - The near clip plane of the camera in units of meters.

Return Values

None.

Description

Sets the camera's near clip plane. Units are specified in metres. The default near clip plane for a newly created camera is 0.1. The minimum value that the function will accept is 0.001.

Examples

```
Camera.SetViewPortZNear( gameCamera, 0.2 )
```

See Also

- Camera.GetViewPortZNear

Cloth

Cloth.SetWindAngle

Brief

Set the wind angle for all cloth meshes.

Definition

```
Cloth.SetWindAngle(number angle)
```

Arguments

angle - The wind angle in degrees.

Return Values

None.

Description

Set the angle for the wind simulation applied to cloth meshes. The angle is measured such that 0 degrees points down the positive X axis, and moves anti-clockwise so that 90 is +Y, 180 is -X and 270 is -Y. All cloth meshes have the same wind angle.

Examples

```
local cloth = myEntity:GetCloth()
if (cloth) then
    cloth:SetWindAngle(135)
end
```

See Also

- Entity.GetCloth

Clubs

Clubs.GetClubCount

Brief

Returns the number of clubs the current user is associated with.

Definition

```
number Clubs.GetClubCount()
```

Arguments

None

Return Values

The number of clubs.

Description

Returns The number of clubs the current user belongs to, or is associated with via invites or requests. Users are listed as "members" of clubs that they are currently invited to or have requested membership to, even if the invite or request has not been accepted. In this case, the member's status field will indicate this.

Examples

```
local numClubs = Clubs.GetClubCount()
print('The user belongs to ' .. numClubs .. ' clubs')
```

See Also

Clubs.GetClubInfo

Brief

Returns the full data relating to a specified club to which the user belongs.

Definition

```
table Clubs.GetClubInfo(number index | string clubId)
```

Arguments

index - The index of which club to get information from. Indicies start at 1.clubId - The club ID to get club information from. This can be retrieved from Clubs.GetClubList.

Return Values

Information about the specified club in the form of a table, or nil if the clubId is not accessible to this user.

Description

Retrieves the full club data for a specified club that the user belongs to. The format of the table is:
{
string id,
string name,
string tag,
number memberCount,
ClubRole role,
ClubMemberStatus status,
Time dateCreated,
string description
}

The role can be one of the following values:
ClubRole.Unknown
ClubRole.NonMember
ClubRole.Member
ClubRole.Subleader
ClubRole.Leader

The status can be one of the following values:
ClubMemberStatus.Member
ClubMemberStatus.Invited
ClubMemberStatus.Requested
The dateCreated is the date/time of the club creation, in server time in UTC. The status and role are for the current user. The club membership data may be a maximum of 1 hour out of date, as it is based on a local cache of the PSN club data, which is periodically refreshed. All data is read only. Only club IDs obtained from Clubs.GetClubList() are valid, any others will generate a Lua error.

Examples

```
local clubs = Clubs.GetClubList()
print('User is in ' .. #clubs .. ' clubs:')
for i, club in ipairs(clubs) do
    local info = Clubs.GetClubInfo(club.id)
    print('Club ' .. club.name .. ' description: ' .. info.description)
end
```

See Also

- Clubs.GetClubList
- Clubs.GetMemberList
- Clubs.GetMemberInfo

Clubs.GetClubList

Brief

Returns the list of Home clubs to which the current user belongs.

Definition

```
table[] Clubs.GetClubList()
```

Arguments

None

Return Values

The information as an array (integer-indexed table) of tables.

Description

Obtains the list of Home clubs to which the current user belongs, returned as an array of tables. Each entry in the array is of the following form:
{
string id,
string name,
string tag,
number memberCount,
ClubRole role,
ClubMemberStatus status
}

The role can be one of the following values:
ClubRole.Unknown
ClubRole.NonMember
ClubRole.Member
ClubRole.Subleader
ClubRole.Leader

The status can be one of the following values:
ClubMemberStatus.Member
ClubMemberStatus.Invited
ClubMemberStatus.Requested
The club membership data may be a maximum of 1 hour out of date, as it is based on a local cache of the PSN club data, which is periodically refreshed. All data is read only. The status and role are for the current user.

Examples

```
local clubs = Clubs.GetClubList()
print('User is in ' .. #clubs .. ' clubs:')
for i, club in ipairs(clubs) do
    print('Club ' .. club.id .. ' ' .. club.name)
end
```

See Also

- Clubs.GetClubInfo
- Clubs.GetMemberList
- Clubs.GetMemberInfo

Clubs.GetInstanceMemberList

Brief

Returns the list of members of a specified club that are within the current instance.

Definition

```
table[] Clubs.GetInstanceMemberList(number index | string clubId)
```

Arguments

index - The index of which club to get information from. Indicies start at 1.clubId - The club ID to get club information from. This can be retrieved from Clubs.GetClubList.

Return Values

The information as an array (integer-indexed table) of tables, or nil if the clubId is not accessible to this user.

Description

Obtains the list of club members within the current instance at the time of calling, returned as an array of tables. The local player is implicit and is not included in the list. Each entry in the arry is of the following form:
{
Person person,
ClubRole role,
ClubMemberStatus status
}

The role can be one of the following values:
ClubRole.Unknown
ClubRole.NonMember
ClubRole.Member
ClubRole.Subleader
ClubRole.Leader

The status can be one of the following values:
ClubMemberStatus.Member
ClubMemberStatus.Invited
ClubMemberStatus.Requested
The member list data may be a maximum of 1 hour out of date, as it is based on a local cache of the PSN club data, which is periodically refreshed. All data is read only. Only club IDs obtained from Clubs.GetClubList() are valid, any others will generate a Lua error. Note that users are listed as "members" of clubs that they are currently invited to or have requested membership to, even if the invite or request has not been accepted. In this case, the member's status field will indicate this. Only users who are full members of the specified club will be able to retrieve the member list, for invited or pending users this function will return nil.

Examples

```
local clubs = Clubs.GetClubList()
local members = Clubs.GetInstanceMemberList(clubs[1].id)
for j, member in ipairs(members) do
    print('  ' .. member.user:GetName())
end
```

See Also

- Clubs.GetClubList
- Clubs.GetMemberInfo
- Clubs.GetMemberList

Clubs.GetMemberCount

Brief

Obtain the number of members in a given club.

Definition

```
number Clubs.GetMemberCount(number index | string clubId)
```

Arguments

index - club indexclubId - club ID obtained from GetClubList

Return Values

The number of members in the club, or nil if the clubId is not accessible to this user.

Description

If the user is not a full member of the club, this function will return the number of full members of the club, not including any pending or invited members. If the user is a full member of the club it returns the total number of users that are associated with the club. This includes users that have been invited or have requested to join, but are not full members.

Examples

```
if (Clubs.GetMemberCount(0) == 32) then
    print("Club 0 is full!")
end
```

See Also

- Clubs.GetClubList

Clubs.GetMemberInfo

Brief

Returns the member data, if any, relating to a specified user/person in a club the they belongs to.

Definition

```
table Clubs.GetMemberInfo(string clubId, User|Person user)
```

Arguments

clubId - the ID of the club to retrieve the member info foruser - a User or Person object identifying the user to retrieve membership info for

Return Values

The information as a table or nil if the user is not associated with the club, or nil if the clubId is not accessible to this user.

Description

Returns the member data, if any, relating to a specified user in a club the user belongs to. If the user is a member of the club, a table of the following form is returned:
{
ClubRole role,
ClubMemberStatus status
}

The role can be one of the following values:
ClubRole.Unknown
ClubRole.NonMember
ClubRole.Member
ClubRole.Subleader
ClubRole.Leader

The status can be one of the following values:
ClubMemberStatus.Member
ClubMemberStatus.Invited
ClubMemberStatus.Requested
The club membership data may be a maximum of 1 hour out of date, as it is based on a local cache of the PSN club data, which is periodically refreshed. All data is read only. Only club IDs obtained from Clubs.GetClubList() are valid, any others will generate a Lua error. Only users who are full members of the specified club will be able to retrieve the member info, for invited or pending users this function will return nil.

Examples

```
local clubs = Clubs.GetClubList()
if (#clubs > 0) then
    local friends = LocalPlayer.GetFriends()
    print('The following of your friends also belong to the ' .. clubs[1].name .. ' club:')
    for i, friend in ipairs(friends) do
        local memInfo = Clubs.GetMemberInfo(clubs[1].id, friend)
        if (memInfo ~= nil and memInfo.status == ClubMemberStatus.Member) then
            print('  ' .. friend:GetName())
        end
    end
end
```

See Also

Clubs.GetMemberList

Brief

Returns the member list of a specified club to which the current user belongs.

Definition

```
table[] Clubs.GetMemberList(number index | string clubId)
```

Arguments

None

Return Values

The information as an array (integer-indexed table) of tables, , or nil if the clubId is not accessible to this user.

Description

Obtains the member list of a club to which the current user belongs, returned as an array of tables. Each entry in the array is of the following form:
{
User user,
ClubRole role,
ClubMemberStatus status
}

The role can be one of the following values:
ClubRole.Unknown
ClubRole.NonMember
ClubRole.Member
ClubRole.Subleader
ClubRole.Leader

The status can be one of the following values:
ClubMemberStatus.Member
ClubMemberStatus.Invited
ClubMemberStatus.Requested
The member list data may be a maximum of 1 hour out of date, as it is based on a local cache of the PSN club data, which is periodically refreshed. All data is read only. Only club IDs obtained from Clubs.GetClubList() are valid, any others will generate a Lua error. Note that users are listed as "members" of clubs that they are currently invited to or have requested membership to, even if the invite or request has not been accepted. In this case, the member's status field will indicate this. Only users who are full members of the specified club will be able to retrieve the member list, for invited or pending users this function will return nil.

Examples

```
local clubs = Clubs.GetClubList()
for i, club in ipairs(clubs) do
    print('Members of ' .. club.name .. ':')
    local members = Clubs.GetMemberList(club.id)
    for j, member in ipairs(members) do
        print('  ' .. member.user:GetName())
    end
end
```

See Also

- Clubs.GetClubList
- Clubs.GetClubInfo
- Clubs.GetMemberInfo

Clubs.IsRefreshed

Brief

Returns true if the club data is up-to-date and false otherwise.

Definition

```
boolean Clubs.IsRefreshed()
```

Arguments

None

Return Values

true if the club data is up-to-date, false if data is not up-to-date or is currently refreshing.

Description

Checks if the club data is up-to-date and ready to be accessed. If a refresh has been requested, this function will return false, and if the data is out-of-date it will also return false.

Examples

```
if (Clubs.RequestRefresh()) then
    state = RefreshState.Refreshing     -- custom user "enum"
end
if (state == RefreshState.Refreshing and Clubs.IsRefreshed()) then
    print('Club data up to date')
    state = RefreshState.Idle
end
```

See Also

- Clubs.RequestRefresh

Clubs.IsRefreshing

Brief

Returns true if the club data is currently refreshing.

Definition

```
boolean Clubs.IsRefreshing()
```

Arguments

None

Return Values

true if the club data is refreshing, false if the club data system is idle.

Description

Checks if the club data is currently refreshing. If a refresh has been requested, this function will return true until the data is full refreshed. The old club data is still valid while the refresh is in progress.

Examples

```
if (Clubs.IsRefreshing()) then
    ShowRefreshingClubsIcon()       -- user function
end
```

See Also

- Clubs.RequestRefresh
- Clubs.IsRefreshed

Clubs.RequestRefresh

Brief

Request an update of the club data.

Definition

```
boolean Clubs.RequestRefresh()
```

Arguments

None

Return Values

true if the refresh was started, false if the club data is already up-to-date.

Description

Requests a refresh of the local club data cache. If false is returned, the club data is already up-to-date and can be used by the script. The current club data remains accessible while the refresh is taking place and is atomically replaced once complete, so scripts can rely on the data being consistent during the refresh.

Examples

```
if (Clubs.RequestRefresh()) then
    state = RefreshState.Refreshing     -- custom user "enum"
end
if (state == RefreshState.Refreshing and Clubs.IsRefreshed()) then
    print('Club data up to date')
    state = RefreshState.Idle
end
```

See Also

- Clubs.IsRefreshed

Collision

Collision.Create

Brief

Creates a new trigger volume.

Definition

```
TriggerVolume Collision.CreateTriggerVolume( string triggerVolumeName, number maxOverlaps, enum
shapeType, number shapeDim1, ... )
TriggerVolume Collision.CreateTriggerVolume( string triggerVolumeName, number maxOverlaps, string
shapeResourceName )
TriggerVolume Collision.CreateTriggerVolume( string triggerVolumeName, number maxOverlaps, Resource
shapeResource )
```

Arguments

triggerVolumeName - The name to give to the trigger volume.maxOverlaps - The maximum number of overlaps to process.shapeType - The TriggerVolumeShape enum defining the shape of the trigger volume.shapeDimN - The Nth dimension of the shape of the trigger volume.shapeResourceName - The name of the collision resource representing the shape of the trigger volume.shapeResource - The collision resource representing the shape of the trigger volume.

Return Values

A new TriggerVolume instance.

Description

Creates a new trigger volume. Valid values for shapeType are:
TriggerVolumeShape.Sphere
TriggerVolumeShape.Box
TriggerVolumeShape.Cylinder
TriggerVolumeShape.Capsule

Examples

```
function MyTriggerVolumeCallback1( triggerVolume, state, count )
    if state == TriggerVolumeOverlapState.Enter then
        print( count .. " new overlaps for trigger volume " .. tostring( triggerVolume ) .. " this
frame" )
    elseif state == TriggerVolumeOverlapState.Exit then
        print( count .. " old overlaps for trigger volume " .. tostring( triggerVolume ) .. " this
frame" )
    end
end

function MyTriggerVolumeCallback2( triggerVolume, state, count )
    for i = 1, count do
        local   overlapType = TriggerVolume.GetOverlapType( triggerVolume, i )
        local   overlapData = TriggerVolume.GetOverlapData( triggerVolume, i )
        print( "trigger volume " .. tostring( triggerVolume ) .. " overlap " .. tostring( i ) .. ":
type=" .. tostring( overlapType ) .. ", object=" .. tostring( overlapData.object ) )
    end
end

triggerVolumes = {}
triggerVolumes[1] = Collision.CreateTriggerVolume( "simpleTriggerVolume1", 16,
TriggerVolumeShape.Sphere, 1.0 )
triggerVolumes[2] = Collision.CreateTriggerVolume( "simpleTriggerVolume2", 16, TriggerVolumeShape.Box,
3.0, 2.0, 4.0 )
triggerVolumes[3] = Collision.CreateTriggerVolume( "resourceTriggerVolume1", 16,
"kTriggerVolumeShapeResource1" )
triggerVolumes[4] = Collision.CreateTriggerVolume( "resourceTriggerVolume2", 16, Resource.Find(
"kTriggerVolumeShapeResource2" ) )

TriggerVolume.SetLocalMatrix( triggerVolumes[1], triggerVolumeTransformMatrix[1] )
TriggerVolume.SetLocalMatrix( triggerVolumes[2], TriggerVolume.GetLocalMatrix( triggerVolumes[1] ) )
TriggerVolume.SetPosition( triggerVolumes[3], triggerVolumePosition[1] )
TriggerVolume.SetPosition( triggerVolumes[4], TriggerVolume.GetPosition( triggerVolumes[3] ) )

for i = 1, #triggerVolumes do
    TriggerVolume.AddCallback( triggerVolumes[i], MyTriggerVolumeCallback1 )
    TriggerVolume.AddCallback( triggerVolumes[i], MyTriggerVolumeCallback2 )
    if TriggerVolume.IsLayerOverlapsEnabled( triggerVolumes[i], CollisionLayer.Scene ) then
        TriggerVolume.SetLayerOverlapsEnabled( triggerVolumes[i], CollisionLayer.Scene, false )
    end
    if not TriggerVolume.IsEnabled( triggerVolumes[i] ) then
        TriggerVolume.SetEnabled( triggerVolumes[i], true )
    end
end

TriggerVolume.RemoveCallback( triggerVolumes[3], MyTriggerVolumeCallback2 )
TriggerVolume.RemoveAllCallbacks( triggerVolumes[4] )

local   numOverlaps = TriggerVolume.GetOverlapCount( triggerVolumes[1] )

for i = 1, numOverlaps do
    local   overlapType = TriggerVolume.GetOverlapType( triggerVolumes[1], i )
    local   overlapData = TriggerVolume.GetOverlapData( triggerVolumes[1], i )
    print( "trigger volume " .. tostring( triggerVolumes[1] ) .. " overlap " .. tostring( i ) .. ":
type=" .. tostring( overlapType ) .. ", object=" .. tostring( overlapData.object ) )
end
```

See Also

None

Collision.CreateLinearCast

Brief

Create a linear cast object of a specified shape.

Definition

```
LinearCast Collision.CreateLinearCast(ShapeType|string|Resource shape, ..., number maxOverlaps = 1)
```

Arguments

shape - Either a ShapeType enum value, or a string or resource representing a collision shape.
If a ShapeType is specified, the value defines the immediately following parameters:
ShapeType.Sphere: number radius - sphere radius.
ShapeType.Box: Vector4 halfExtents - X, Y, Z half-extents.
ShapeType.Capsule: Vector4 point1, Vector4 point2, number radius - capsule points and radius.maxOverlaps - Maximum number of overlaps to store, must be at least 1.

Return Values

A LinearCast object if successful, or nil otherwise.

Description

A linear cast will sweep a shape through a physics world and return collision items that overlap the swept volume. Hits will be stored until maxOverlaps hits have been recorded, at which point the sweet will stop. More complex shapes will be less efficient to sweep through the world.

Examples

```
cast = Collision.CreateLinearCast(ShapeType.Box, Vector4.Create(0.2, 0.8, 0.2), 5)      -- create cast
of box of half-extents [0.2, 0.8, 0.2], 5 max overlaps
rot = Matrix44.Create()
rot:SetRotationXyz(0, 45, 0)
cast:SetRotation(rot)
cast:Start(Vector4.Create(0,0,0), Vector4.Create(5,0,0))
while (not cast:IsReady()) do
    coroutine.yield()
end
local v = Vector4.Create()
for i = 1, cast:GetHitCount() do
    cast:GetHitPoint(i, v)
    print("Hit point:", v)
    cast:GetHitNormal(i, v)
    print("Hit normal:", v)
    local type = cast:GetHitType(i)
    print("Hit type:", type)
    print("Hit distance:", cast:GetHitDistance(i))
    local data = cast:GetHitData(i)
    if (type == HitType.Entity) then
        -- data is an entity
    elseif (type == HitType.Person) then
        -- data is a person
    elseif (type == HitType.Furniture) then
        -- data is a table (see GetHitData)
    elseif (type == HitType.Scene) then
        -- data is user ID of scene feature hit
    else
        -- data is nil - no further info available
    end
end
```

See Also

- LinearCast.Start

Collision.CreateRaycast

Brief

Creates a new Raycast object.

Definition

```
Raycast Collision.CreateRaycast()
```

Arguments

None

Return Values

A new Raycast instance.

Description

Creates a new Raycast object.

Examples

```
raycast = Collision.CreateRaycast()
```

See Also

- Raycast.Start

Debug

Debug.ArcadeGameGetActive

Brief

Returns the object of the currently active arcade game.

Definition

```
Object Debug.ArcadeGameGetActive()
```

Arguments

None

Return Values

The object of the currently active arcade game and nil if there is not one.

Description

Returns the object of the currently active arcade game and nil if there is not one.

Examples

```
arcadeGame = Debug.ArcadeGameGetActive()
print( "The active game is " .. Object.GetObjectId( arcadeGame ) )
```

See Also

- Object.GetObjectId

Debug.ArcadeGameLoad

Brief

Loads the specified arcade game.

Definition

```
Debug.ArcadeGameLoad( string objectId )
```

Arguments

objectId - The object id of the arcade game.

Return Values

None

Description

Loads the specified arcade game.

Examples

```
-- The object ID of the game as shown in Object Editor
myGame = "4626CD25-2B27433A-983EB874-7061BBAB"
Debug.ArcadeGameLoad( myGame )
```

See Also

None

Debug.ArcadeGameReset

Brief

Resets the currently playing arcade game.

Definition

```
Debug.ArcadeGameReset()
```

Arguments

None

Return Values

None

Description

Resets the currently playing arcade game. As part of the reset process, the arcade game object is completely reloaded, including all script

resources. This removes the need for reloading the Home runtime during the development of an arcade game. Typically, this function is called from the debug console (via the 'lc' command), or from a script run from the console via the RunScript function.

Examples

```
function ResetArcadeGameThread()
    while ( true ) do
        if Debug.KeyIsPressed( KEY_F2 ) then
            Debug.ArcadeGameReset()
        end
    end
end

ResetArcadeGameThreadId = SpawnFunction( ResetArcadeGameThread )
```

See Also

- Debug.KeyIsPressed
- Debug.MiniGameReset
- Debug.SceneReset
- Debug.RunScript
- SpawnFunction

Debug.ArcadeGameSetBorderColor

Brief

Sets the color of the arcade game's border when running it fullscreen.

Definition

```
Debug.ArcadeGameSetBorderColor( Vector4 rgb )
```

Arguments

rgb - The color to set the border to.

Return Values

None

Description

Sets the color of the arcade game's left and right borders when running it in fullscreen mode (using the --game command line argument for HomeDeveloper.self). The screen space of a Home arcade game is fixed to 1024x720. However, the screen space of the entire display device is 1280x720 which leaves vertical borders on the left and right hand sides of the screen. By default, the border color is set to magenta.

Examples

```
-- Set border color to green
Debug.ArcadeGameSetBorderColor( Vector4.Create( 0, 1, 0 ) )

-- Set border color to magenta, the default color
Debug.ArcadeGameSetBorderColor( Vector4.Create( 1, 0, 1 ) )
```

See Also

- Vector4.Create

Debug.DebuggerRegister

Brief

Registers the Lua environment with the debugger.

Definition

```
Debug.DebuggerRegister()
```

Arguments

None

Return Values

None

Description

Registers the Lua environment with the debugger. When registered with the debugger you can debug the Lua script using SLED, the HDK's Lua debugger.

Examples

```
Debug.DebuggerRegister()
Debug.DebuggerSetupProject()
```

See Also

- Debug.DebuggerSetupProject
- Debug.DebuggerUnRegister

Debug.DebuggerSetupProject

Brief

Sets up the project for the debugger before connecting to the debugger.

Definition

```
Debug.DebuggerSetupProject( string projectName = "Home" )
```

Arguments

projectName - The name of the debugger project.

Return Values

None

Description

Sets up the project for the debugger before connecting to the debugger. All the files opened in the registered Lua environments are added to the project at the time the function is called.

Examples

```
Debug.DebuggerRegister()
Debug.DebuggerSetupProject()
```

See Also

- Debug.DebuggerRegister
- Debug.DebuggerUnRegister

Debug.DebuggerUnRegister

Brief

Unregisters the Lua environment with the debugger.

Definition

```
Debug.DebuggerUnRegister()
```

Arguments

None

Return Values

None

Description

Unregisters the Lua environment with the debugger. By default the environment is unregistered.

Examples

```
Debug.DebuggerUnRegister()
```

See Also

- Debug.DebuggerRegister
- Debug.DebuggerSetupProject

Debug.DrawBox3d

Brief

Draws a 3-dimensional box in the scene.

Definition

```
Debug.DrawBox3d( Matrix44 transform, Vector4 rgba )
```

Arguments

transform - Transform of box.rgba - The color of the box.

Return Values

None

Description

Draws a 3-dimensional box in the scene. For debug purposes only. The origin is at the center of the box.

Examples

```
matrix = Matrix44.Create()
Matrix44.SetScale( matrix, 1, 0.4, 3, 1 )
Matrix44.SetRotationXyz( matrix, 0, 30, 18, 0 )
Matrix44.SetTranslation( matrix, 0, 2, 0, 0 )
Debug.DrawBox3d( matrix, Vector4.Create( 1, 0, 1, 0.5 ) )
```

See Also

- Debug.DrawSphere3d
- Debug.DrawLine3d
- Debug.DrawCone3d
- Debug.DrawCylinder3d

Debug.DrawCone3d

Brief

Draws a 3-dimensional Cone in the scene.

Definition

```
Debug.DrawCone3d( Matrix44 transform, Vector4 rgba )
```

Arguments

transform - Transform of Cone.rgba - The color of the cone.

Return Values

None

Description

Draws a 3-dimensional Cone in the scene. For debug purposes only. The origin is the top of the cone and the cone points down the Z axis.

Examples

```
matrix = Matrix44.Create()
Matrix44.SetScale( matrix, 1, 0.4, 3, 1 )
Matrix44.SetRotationXyz( matrix, 0, 30, 18, 0 )
Matrix44.SetTranslation( matrix, 0, 2, 0, 0 )
Debug.DrawCone3d( matrix, Vector4.Create( 1, 0, 1, 0.5 ) )
```

See Also

- Debug.DrawSphere3d
- Debug.DrawLine3d

- Debug.DrawBox3d
- Debug.DrawCylinder3d

Debug.DrawCylinder3d

Brief

Draws a 3-dimensional Cylinder in the scene.

Definition

```
Debug.DrawCylinder3d( Matrix44 transform, Vector4 rgba )
Debug.DrawCylinder3d( Vector4 point0, Vector4 point1, number radius, Vector4 rgba )
```

Arguments

transform - Transform of Cylinder.rgba - The color of the cylinder.point0 - One end of the cylinder axispoint1 - Opposite end of the cylinder axisradius - Radius of the cylinder

Return Values

None

Description

Draws a 3-dimensional Cylinder in the scene. For debug purposes only. The origin is the center of its first side.

Examples

```
matrix = Matrix44.Create()
Matrix44.SetScale( matrix, 1, 0.4, 3, 1 )
Matrix44.SetRotationXyz( matrix, 0, 30, 18, 0 )
Matrix44.SetTranslation( matrix, 0, 2, 0, 0 )
Debug.DrawCylinder3d( matrix, Vector4.Create( 1, 0, 1, 0.5 ) )
```

See Also

- Debug.DrawSphere3d
- Debug.DrawLine3d
- Debug.DrawCone3d
- Debug.DrawBox3d

Debug.DrawLine3d

Brief

Draws a 3-dimensional line in the scene.

Definition

```
Debug.DrawLine3d( Vector4 start, Vector4 end, Vector4 rgba )
```

Arguments

start - The first point of the line.end - The last point of the line.rgba - The color of the line.

Return Values

None

Description

Draws a 3-dimensional line in the scene. For debug purposes only.

Examples

```
Debug.DrawLine3d( Vector4.Create( 0, 0, 0 ), Vector4.Create( 10, 10, 0 ), Vector4.Create( 1, 0, 0, 1 )
)
```

See Also

- Debug.DrawRect
- Debug.DrawText

Debug.DrawQuad3d

Brief

Draws a 3-dimensional quad in the scene.

Definition

```
Debug.DrawQuad3d( Vector4 v1, Vector4 v2, Vector4 v3, Vector4 v4, Vector4 rgba, boolean isWireFrame =
false )
```

Arguments

v1 - First quad vertex.v2 - Second quad vertex.v3 - Third quad vertex.v4 - Fourth quad vertex.rgba - The color of the quad.isWireFrame - Render quad wire-frame or solid (default).

Return Values

None

Description

Draws a 3-dimensional quad in the scene. For debug purposes only.

Examples

```
-- Draw a solid red quad
v1 = Vector4.Create( 0, 0, 0 )
v2 = Vector4.Create( 10, 0, 0 )
v3 = Vector4.Create( 10, 10, 0 )
v4 = Vector4.Create( 0, 10, 0 )
rgba = Vector4.Create( 1, 0, 0, 1 )
Debug.DrawQuad3d( v1, v2, v3, v4, rgba, false )
```

See Also

- Debug.DrawSphere3d
- Debug.DrawLine3d
- Debug.DrawCone3d

- Debug.DrawBox3d
- Debug.DrawCylinder3d
- Debug.DrawTriangle3d

Debug.DrawRect

Brief

Debug function used to display a quad on the 2d overlay.

Definition

```
Debug.DrawRect( number x, number y, number w, number h, rgba = [1, 1, 1, 1] )
```

Arguments

x - The x-component of the top left screen space coordinate of the quad.y - The y-component of the top left screen space coordinate of the quad.w - The width in pixels of the quad.h - The height in pixels of the quad.rgba - The color for the rectangle.

Return Values

None

Description

This function is available to aid debugging, allowing the LUA script to display untextured quads on the main 2d overlay.

Examples

```
-- Render a 100x100 square in the top left corner of the screen
Debug.DrawRect( 0, 0, 100, 100 )
```

See Also

- Debug.DrawLine3d
- Debug.DrawText

Debug.DrawSphere3d

Brief

Draws a 3-dimensional Sphere in the scene.

Definition

```
Debug.DrawSphere3d( Matrix44 transform, Vector4 rgba )
Debug.DrawSphere3d( Vector4 center, number radius, Vector4 rgba )
```

Arguments

transform - Transform of the sphere.center - Center point of the sphere.radius - Sphere radius.rgba - The color of the sphere.

Return Values

None

Description

Draws a 3-dimensional Sphere in the scene. For debug purposes only. The origin is the center of the sphere.

Examples

```
matrix = Matrix44.Create()
Matrix44.SetScale( matrix, 1, 0.4, 3, 1 )
Matrix44.SetRotationXyz( matrix, 0, 30, 18, 0 )
Matrix44.SetTranslation( matrix, 0, 2, 0, 0 )
Debug.DrawSphere3d( matrix, Vector4.Create( 1, 0, 1, 0.5 ) )
```

See Also

- Debug.DrawLine3d
- Debug.DrawCone3d
- Debug.DrawBox3d
- Debug.DrawCylinder3d

Debug.DrawText

Brief

Debug function used to display text on the 2d overlay

Definition

```
Debug.DrawText( number x, number y, string message )
```

Arguments

x - Screen space x-coordinate marking the start position of the string.y - Screen space y-coordinate marking the start position of the string.message - Text to display.

Return Values

None

Description

This function is available to aid debugging, allowing the LUA script to display text on the main 2d overlay. The rendered text is near-white with a black drop shadow.

Examples

```
Debug.DrawText(200, 200, "Hello World!" )
```

See Also

- Debug.DrawLine3d
- Debug.DrawRect

Debug.DrawTriangle3d

Brief

Draws a 3-dimensional triangle in the scene.

Definition

```
Debug.DrawTriangle3d( Vector4 v1, Vector4 v2, Vector4 v3, Vector4 rgba, boolean isWireFrame = false )
```

Arguments

v1 - First triangle vertex.v2 - Second triangle vertex.v3 - Third triangle vertex.rgba - The color of the triangle.isWireFrame - Render triangle wire-frame or solid (default).

Return Values

None

Description

Draws a 3-dimensional triangle in the scene. For debug purposes only.

Examples

```
-- Draw a solid red triangle
v1 = Vector4.Create( 0, 0, 0 )
v2 = Vector4.Create( 10, 0, 0 )
v3 = Vector4.Create( 0, 10, 0 )
rgba = Vector4.Create( 1, 0, 0, 1 )
Debug.DrawTriangle3d( v1, v2, v3, rgba, false )
```

See Also

- Debug.DrawSphere3d
- Debug.DrawLine3d
- Debug.DrawCone3d
- Debug.DrawBox3d
- Debug.DrawCylinder3d
- Debug.DrawQuad3d

Debug.GetDevMode

Brief

Returns the user defined dev mode.

Definition

```
string Debug.GetDevMode()
```

Arguments

None

Return Values

The user defined dev mode.

Description

Returns the user defined dev mode specified with the command line argument --devmode such as --devmode lua_debug.

Examples

```
function Update()
    if ( Debug.GetDevMode() == "lua_debug" ) then
        print( "a = " .. a .. " b = " .. b )
    end
end
```

See Also

None

Debug.GetStack

Brief

Returns the stack as a string.

Definition

```
string Debug.GetStack()
```

Arguments

None

Return Values

The stack as a string.

Description

Returns the stack as a string. For debug purposes only.

Examples

```
print( "The stack at this point is " .. Debug.GetStack() )
```

See Also

None

Debug.GetThreadIds

Brief

Returns a table of thread IDs for all the Lua threads.

Definition

```
number[] Debug.GetThreadIds()
```

Arguments

None

Return Values

A table of numbers representing thread IDs.

Description

Returns a table of thread IDs for all the Lua threads.

Examples

```
ids = Debug.GetThreadIds()
```

See Also

None

Debug.GetWorldToScreen

Brief

Converts a 3d world position to a screen position.

Definition

```
Vector4 Debug.GetWorldToScreen( Vector4 worldPosition )
```

Arguments

worldPosition - The 3d position in the world.

Return Values

The screen space coordinate in the range [0,0] -> [1280,720].

Description

Converts a 3d world position to a screen position.

Examples

```
screenPos = Debug.GetWorldToScreen( worldPos )
```

See Also

None

Debug.KeyIsHeld

Brief

Returns true if the specified key is held and false otherwise.

Definition

```
boolean Debug.KeyIsHeld( number key )
```

Arguments

key - The key to test. Can be one of the following:

Function keys:
KEY_F1 to KEY_F24

Alpabetic keys:
KEY_A to KEY_Z

Numeric keys:
KEY_0 to KEY_9

Number pad numeric keys:
KEY_NUM0 to KEY_NUM9

Miscellaneous keys:
KEY_ESCAPE, KEY_GRAVE, KEY_MINUS, KEY_EQUALS,
KEY_BACK, KEY_TAB, KEY_LEFTBRACKET, KEY_RIGHTBRACKET,
KEY_RETURN, KEY_CAPSLOCK, KEY_SEMICOLON, KEY_QUOTE,
KEY_CONSOLE, KEY_LEFTSHIFT, KEY_BACKSLASH, KEY_COMMA,
KEY_PERIOD, KEY_FORWARDSLASH, KEY_RIGHTSHIFT,
KEY_LEFTCONTROL, KEY_LEFTWINDOWS, KEY_LEFTALT,
KEY_SPACE, KEY_RIGHTALT, KEY_RIGHTWINDOWS,
KEY_RIGHTCONTROL, KEY_PRINTSCREEN, KEY_SCROLLLOCK,
KEY_PAUSE, KEY_HOME, KEY_PAGEUP, KEY_END, KEY_PAGEDOWN,
KEY_DELETE, KEY_INSERT, KEY_UP, KEY_LEFT, KEY_DOWN,
KEY_RIGHT, KEY_NUMLOCK, KEY_NUMDIV, KEY_NUMMUL,
KEY_NUMSUB, KEY_NUMADD, KEY_NUMENTER, KEY_NUMDEL,
KEY_APP, KEY_POWER, KEY_NUMEQUALS, KEY_EXECUTE,
KEY_HELP, KEY_MENU, KEY_SELECT, KEY_STOP1, KEY_AGAIN,
KEY_UNDO, KEY_CUT, KEY_COPY, KEY_PASTE, KEY_FIND,
KEY_MUTE1, KEY_VOLUMEUP1, KEY_VOLUMEDOWN1,
KEY_NEXTTRACK, KEY_PREVTRACK, KEY_STOP2, KEY_PLAY,
KEY_MUTE2, KEY_VOLUMEUP2, KEY_VOLUMEDOWN2

Return Values

true if the specified key is held and false otherwise.

Description

Returns true if the specified key is held and false otherwise.

Examples

```
print( "Escape held = " .. Debug.KeyIsHeld( KEY_ESCAPE ) )
```

See Also

- Debug.KeyIsPressed
- Debug.KeyIsReleased

Debug.KeyIsPressed

Brief

Returns true if the specified key is pressed and false otherwise.

Definition

```
boolean Debug.KeyIsPressed( number key )
```

Arguments

key - The key to test. Can be one of the following:

Function keys:
KEY_F1 to KEY_F24

Alpabetic keys:
KEY_A to KEY_Z

Numeric keys:
KEY_0 to KEY_9

Number pad numeric keys:
KEY_NUM0 to KEY_NUM9

Miscellaneous keys:
KEY_ESCAPE, KEY_GRAVE, KEY_MINUS, KEY_EQUALS,
KEY_BACK, KEY_TAB, KEY_LEFTBRACKET, KEY_RIGHTBRACKET,
KEY_RETURN, KEY_CAPSLOCK, KEY_SEMICOLON, KEY_QUOTE,
KEY_CONSOLE, KEY_LEFTSHIFT, KEY_BACKSLASH, KEY_COMMA,
KEY_PERIOD, KEY_FORWARDSLASH, KEY_RIGHTSHIFT,
KEY_LEFTCONTROL, KEY_LEFTWINDOWS, KEY_LEFTALT,
KEY_SPACE, KEY_RIGHTALT, KEY_RIGHTWINDOWS,
KEY_RIGHTCONTROL, KEY_PRINTSCREEN, KEY_SCROLLLOCK,
KEY_PAUSE, KEY_HOME, KEY_PAGEUP, KEY_END, KEY_PAGEDOWN,
KEY_DELETE, KEY_INSERT, KEY_UP, KEY_LEFT, KEY_DOWN,
KEY_RIGHT, KEY_NUMLOCK, KEY_NUMDIV, KEY_NUMMUL,
KEY_NUMSUB, KEY_NUMADD, KEY_NUMENTER, KEY_NUMDEL,
KEY_APP, KEY_POWER, KEY_NUMEQUALS, KEY_EXECUTE,
KEY_HELP, KEY_MENU, KEY_SELECT, KEY_STOP1, KEY_AGAIN,
KEY_UNDO, KEY_CUT, KEY_COPY, KEY_PASTE, KEY_FIND,
KEY_MUTE1, KEY_VOLUMEUP1, KEY_VOLUMEDOWN1,
KEY_NEXTTRACK, KEY_PREVTRACK, KEY_STOP2, KEY_PLAY,
KEY_MUTE2, KEY_VOLUMEUP2, KEY_VOLUMEDOWN2

Return Values

true if the specified key is pressed and false otherwise.

Description

Returns true if the specified key is pressed and false otherwise. Note, the keyboard is set to text entry mode and key repeat will kick in if a key is held down for too long. Use Debug.SetKeyMode to change this behaviour.

Examples

```
print( "Escape pressed = " .. Debug.KeyIsPressed( KEY_ESCAPE ) )
```

See Also

- Debug.KeyIsHeld
- Debug.KeyIsReleased

Debug.KeyIsReleased

Brief

Returns true if the specified key is released and false otherwise.

Definition

```
boolean Debug.KeyIsReleased( number key )
```

Arguments

key - The key to test. Can be one of the following:

Function keys:
KEY_F1 to KEY_F24

Alpabetic keys:
KEY_A to KEY_Z

Numeric keys:
KEY_0 to KEY_9

Number pad numeric keys:
KEY_NUM0 to KEY_NUM9

Miscellaneous keys:
KEY_ESCAPE, KEY_GRAVE, KEY_MINUS, KEY_EQUALS,
KEY_BACK, KEY_TAB, KEY_LEFTBRACKET, KEY_RIGHTBRACKET,
KEY_RETURN, KEY_CAPSLOCK, KEY_SEMICOLON, KEY_QUOTE,
KEY_CONSOLE, KEY_LEFTSHIFT, KEY_BACKSLASH, KEY_COMMA,
KEY_PERIOD, KEY_FORWARDSLASH, KEY_RIGHTSHIFT,
KEY_LEFTCONTROL, KEY_LEFTWINDOWS, KEY_LEFTALT,
KEY_SPACE, KEY_RIGHTALT, KEY_RIGHTWINDOWS,
KEY_RIGHTCONTROL, KEY_PRINTSCREEN, KEY_SCROLLLOCK,
KEY_PAUSE, KEY_HOME, KEY_PAGEUP, KEY_END, KEY_PAGEDOWN,
KEY_DELETE, KEY_INSERT, KEY_UP, KEY_LEFT, KEY_DOWN,
KEY_RIGHT, KEY_NUMLOCK, KEY_NUMDIV, KEY_NUMMUL,
KEY_NUMSUB, KEY_NUMADD, KEY_NUMENTER, KEY_NUMDEL,
KEY_APP, KEY_POWER, KEY_NUMEQUALS, KEY_EXECUTE,
KEY_HELP, KEY_MENU, KEY_SELECT, KEY_STOP1, KEY_AGAIN,
KEY_UNDO, KEY_CUT, KEY_COPY, KEY_PASTE, KEY_FIND,
KEY_MUTE1, KEY_VOLUMEUP1, KEY_VOLUMEDOWN1,
KEY_NEXTTRACK, KEY_PREVTRACK, KEY_STOP2, KEY_PLAY,
KEY_MUTE2, KEY_VOLUMEUP2, KEY_VOLUMEDOWN2

Return Values

true if the specified key is released and false otherwise.

Description

Returns true if the specified key is released and false otherwise.

Examples

```
print( "Escape released = " .. Debug.KeyIsReleased( KEY_ESCAPE ) )
```

See Also

- Debug.KeyIsHeld
- Debug.KeyIsPressed

Debug.MemGetLargestBlock

Brief

Retrieves the largest block available within the budget.

Definition

```
Debug.MemGetLargestBlock( MemoryType memoryType = MemoryType.Main )
```

Arguments

memoryType - The memory type to check for this budget. Can be one of the following:

Memory Type:
MemoryType.Main
MemoryType.Host
MemoryType.Vram

Return Values

Size in kilobytes (KB) or 0 if there was an error.

Description

Gets the largest available memory block. This isn't guaranteed to be the size of an allocatable block due to alignment restrictions and headers used within the memory system. This is intended to be used as an indication of fragmentation within the budget.

Examples

```
block = Debug.MemGetLargestBlock( MemoryType.Main )
```

See Also

- Debug.MemGetTotal
- Debug.MemGetUsed

Debug.MemGetTotal

Brief

Retrieves the total size of the budget the object is using.

Definition

```
Debug.MemGetTotal( MemoryType memoryType = MemoryType.Main )
```

Arguments

memoryType - The memory type to check for this budget. Can be one of the following:

Memory Type:
MemoryType.Main
MemoryType.Host
MemoryType.Vram

Return Values

Size in kilobytes (KB) or 0 if there was an error.

Description

Gets the total size of the budget for the specified memory type. Defaults to main memory.

Examples

```
total = Debug.MemGetTotal( MemoryType.Main )
```

See Also

- Debug.MemGetUsed
- Debug.MemGetLargestBlock

Debug.MemGetUsed

Brief

Retrieves the amount of memory currently being used.

Definition

```
Debug.MemGetUsed( MemoryType memoryType = MemoryType.Main )
```

Arguments

memoryType - The memory type to check for this budget. Can be one of the following:

Memory Type:
MemoryType.Main
MemoryType.Host
MemoryType.Vram

Return Values

Size in kilobytes (KB) or 0 if there was an error.

Description

Gets the amount of used memory.

Examples

```
used = Debug.MemGetUsed( MemoryType.Main )
```

See Also

- Debug.MemGetTotal
- Debug.MemGetLargestBlock

Debug.MiniGameGetActive

Brief

Returns the mini game the user is joined to and nil if there is not one.

Definition

```
Object Debug.MiniGameGetActive()
```

Arguments

None

Return Values

The mini game the user is joined to and nil if there is not one.

Description

Returns the mini game the user is joined to and nil if there is not one.

Examples

```
miniGame = Debug.MiniGameGetActive()
print( "The active game is " .. Object.GetObjectId( miniGame ) )
```

See Also

- Object.GetObjectId

Debug.MiniGameGetClosest

Brief

Returns the closest mini game and nil if there is not one.

Definition

```
Object Debug.MiniGameGetClosest( boolean onlyJoinable = false )
```

Arguments

onlyJoinable - If true it only returns the closest joinable game.

Return Values

The closest mini game and nil if there is not one.

Description

Returns the closest mini game and nil if there is not one.

Examples

```
miniGame = Debug.MiniGameGetClosest()
print( "The closest game is " .. Object.GetObjectId( miniGame ) )
```

See Also

- Object.GetObjectId

Debug.MiniGameReset

Brief

Resets the closest mini-game to the player's avatar.

Definition

```
Debug.MiniGameReset( string resetType = nil )
```

Arguments

resetType - Can be 'all' or an object ID to reset. If nil it resets the closest game.

Return Values

None

Description

Resets the closest mini-game to the player's avatar. As part of the reset process, the mini-game object is completely reloaded, including all script resources. This removes the need for reloading the Home runtime during the development of a mini-game. Typically, this function is called from the debug console (via the 'lc' command), or from a script run from the console via the RunScript function.

Examples

```
function ResetMiniGameThread()
    while ( true ) do
        if Debug.KeyIsPressed( KEY_F3 ) then
            Debug.MiniGameReset()
        end
    end
end

ResetMiniGameThreadId = SpawnFunction( ResetMiniGameThread )
```

See Also

- Debug.ArcadeGameReset
- Debug.KeyIsPressed
- Debug.SceneReset
- Debug.RunScript
- SpawnFunction

Debug.MouseGetVelocityX

Brief

Returns the horizontal velocity of the mouse.

Definition

```
number Debug.MouseGetVelocityX()
```

Arguments

None

Return Values

The horizontal velocity of the mouse.

Description

Returns the horizontal velocity of the mouse. If the mouse if dragged to the left, it will return a negative number and if it is dragged to the right, it will return a positive number.

Examples

```
mouseX = Debug.MouseGetVelocityX()
```

See Also

- Debug.MouseGetVelocityY
- Debug.MouseIsButtonHeld

Debug.MouseGetVelocityY

Brief

Returns the vertical velocity of the mouse.

Definition

```
number Debug.MouseGetVelocityY()
```

Arguments

None

Return Values

The vertical velocity of the mouse.

Description

Returns the vertical velocity of the mouse. If the mouse is pushed forwards, it will return a negative number and if it is pulled backwards, it will return a positive number.

Examples

```
mouseY = Debug.MouseGetVelocityY()
```

See Also

- Debug.MouseGetVelocityX
- Debug.MouseIsButtonHeld

Debug.MouseIsButtonHeld

Brief

Returns true if the button is held and false otherwise.

Definition

```
boolean Debug.MouseIsButtonHeld( number button )
```

Arguments

button - The button to check if it is held. Can be one of the following:
MOUSE_BUTTON_LEFT
MOUSE_BUTTON_MIDDLE
MOUSE_BUTTON_RIGHT

Return Values

true if the button is held and false otherwise.

Description

Returns true if the button is held and false otherwise.

Examples

```
if ( Debug.MouseIsButtonHeld( MOUSE_BUTTON_RIGHT ) ) then
    print( "Right mouse button held" )
end
```

See Also

- Debug.MouseGetVelocityX
- Debug.MouseGetVelocityY

Debug.MouseIsButtonPressed

Brief

Returns true if the button is pressed and false otherwise.

Definition

```
boolean Debug.MouseIsButtonPressed( number button )
```

Arguments

button - The button to check if it is pressed. Can be one of the following:
MOUSE_BUTTON_LEFT
MOUSE_BUTTON_MIDDLE
MOUSE_BUTTON_RIGHT

Return Values

true if the button is pressed and false otherwise.

Description

Returns true if the button is pressed and false otherwise.

Examples

```
if Debug.MouseIsButtonPressed( MOUSE_BUTTON_RIGHT ) then
    print( "Right mouse button pressed" )
end
```

See Also

- Debug.MouseGetVelocityX

- Debug.MouseGetVelocityY

Debug.MouseIsButtonReleased

Brief

Returns true if the button is released and false otherwise.

Definition

```
boolean Debug.MouseIsButtonReleased( number button )
```

Arguments

button - The button to check if it is released. Can be one of the following:
MOUSE_BUTTON_LEFT
MOUSE_BUTTON_MIDDLE
MOUSE_BUTTON_RIGHT

Return Values

true if the button is released and false otherwise.

Description

Returns true if the button is released and false otherwise.

Examples

```
if Debug.MouseIsButtonReleased( MOUSE_BUTTON_RIGHT ) then
    print( "Right mouse button released" )
end
```

See Also

- Debug.MouseGetVelocityX
- Debug.MouseGetVelocityY

Debug.RunScript

Brief

Loads and runs the specified file.

Definition

```
boolean Debug.RunScript( string fileName )
```

Arguments

fileName - The name of the file to load.

Return Values

None

Description

Loads and runs the specified file. The path is relative to the build folder. If you are running scripts in objects then you must have the Lua files as lua resources and run them with Resource.Run instead.

Examples

```
Debug.RunScript( "myscripts/jumpthread.lua" )
```

See Also

- Resource.Run

Debug.SceneIsReady

Brief

Returns true if the scene is ready and false otherwise.

Definition

```
boolean Debug.SceneIsReady()
```

Arguments

None

Return Values

true if the scene is ready and false otherwise.

Description

Returns true if the scene is ready and false otherwise.

Examples

```
if Debug.SceneIsReady() == true then
    game = Debug.MiniGameGetClosest()
end
```

See Also

None

Debug.SceneReset

Brief

Resets the scene object for dynamic and scripted content.

Definition

```
Debug.SceneReset()
```

Arguments

None

Return Values

None

Description

Resets the scene object for dynamic and scripted content such as scene entity animations and the scene script.

Examples

```
Debug.SceneReset()
```

See Also

- Debug.ArcadeGameReset
- Debug.MiniGameReset

Debug.ScreenSetContent

Brief

Sets a content source or sources on the screen.

Definition

```
Debug.ScreenSetContent( Screen screen, string[] contentList )
Debug.ScreenSetContent( Screen screen, string content, number contentIndex = 1 )
```

Arguments

screen - The screen to update.contentList - An array of strings specifying multiple content sources.content - A string for the content source.contentIndex - The index to set the content to. Ranges from 1 to the number of content sources. Defaults to 1.

Return Values

None.

Description

Sets a content source or sources on the screen. The source string will be a path relative to the HDK's build root directory. Note that paths relative to the HDK's build root directory can only be used for testing/development purposes. Scripts uploaded to a content server must reference content served over HTTP and use the Screen.SetContent function.

Examples

```
-- Example1: Testing a video served from the host PC's hard drive for debug
localSource = "localvideo/testvideo.mp4"
Debug.ScreenSetContent( myScreen1, localSource )

-- Example 2: Play back a video served from an HTTP server
onlineSource = "http://mycontentserver.com/onlinevideo.mp4"
Screen.SetContent( myScreen2, onlineSource )
```

See Also

- Screen.SetContent

Debug.SetErrorCallback

Brief

Specify a function to be called in the event of a script error.

Definition

```
Debug.SetErrorCallback(string callbackFuncName)
```

Arguments

callbackFuncName - Name of global function to call when script environment errors.

Return Values

None.

Description

Sets a function to be called in the event of a script error. During development often a script will contain errors which cause the script to be shut down, and if the script has modified global state it is not given a chance to clean up these changes, which can make re-running the fixed script problematic. If this function is called, it will set the name of a global script function which will be called when the script environment errors, and can be used to perform this clean up. It is only necessary to call this function once upon script initialization, it will be retained until the environment is destroyed. Note that this function is not available outside of developer builds and is not a substitute for proper script clean up in the normal game situations (player leaving game, object destroyed, etc).

Examples

```
Debug.SetErrorCallback("MyErrorFunc")

function MyErrorFunc()
    myCustomCamera:Deactivate()
    myCustomCamera = nil
end
```

See Also

None

Debug.System

Brief

Runs the command string in the console window.

Definition

```
Debug.System( string command )
```

Arguments

command - The command to run.

Return Values

None

Description

Runs the command string in the console window.

Examples

```
Debug.System( "lc print( 'Hello' )" )
```

See Also

None

Entity

Entity.ApplyAngularImpulse

Brief

Applies an angular impulse to the entity.

Definition

```
Entity.ApplyAngularImpulse( Entity entity, Vector4 impulse )
```

Arguments

entity - The entity to update.impulse - World space vector specifying angular impulse to be applied to the entity.

Return Values

None.

Description

Applies an angular impulse to the entity around the center of mass, creating angular velocity.

Examples

```
-- Apply angular impulse around Y axis
Entity.ApplyAngularImpulse( entity, Vector4.Create( 0, 0.1, 0.0) )
```

See Also

None

Entity.ApplyImpulse

Brief

Applies an impulse around a point.

Definition

```
Entity.ApplyImpulse( Entity entity, Vector4 impulse )
Entity.ApplyImpulse( Entity entity, Vector4 impulse, Vector4 point )
```

Arguments

entity - The entity to update.impulse - World space vector specifying impulse (or force) to be applied to the entity.point - World space vector specifying the point around which the impulse acts.

Return Values

None.

Description

Applies an impulse around a point. If no point is specified, the center of mass of the entity's rigid body is used.

Examples

```
-- Apply an impulse around the entity's center of mass
Entity.ApplyImpulse( entity, Vector4.Create( 5, 10, 5 ) )

-- Apply an impulse around the origin of world space
Entity.ApplyImpulse( entity, Vector4.Create( 0, 10, 0 ), Vector4.Create( 0, 0, 0 ) )
```

See Also

None

Entity.AttachPersonLabel

Brief

Attach a person label to the entity.

Definition

```
Entity.AttachPersonLabel(Entity entity, Person person, Vector4 offset)
```

Arguments

person - The person to take the label from.offset - The offset of the label from the entity's position. (Optional. Defaults to 0, 0, 0).

Return Values

Description

This function will detach the person label from the given person, and attach it to the entity instead. Any number of person labels can be attached to the entity.

Examples

```
Entity.AttachPersonLabel(entity, person, labelOffset)
```

See Also

- Entity.DetachPersonLabel

Entity.AttachToBone

Brief

Attach the entity to the specified bone of its parent.

Definition

```
Entity.AttachToBone( Entity entity, string boneName )
```

Arguments

entity - The entity to update.boneName - The name of the bone to attach to or nil to clear it. Can be one of the following:
"pelvis"
"hips"
"leftleg"
"leftfoot"
"rightleg"
"rightfoot"
"spine"
"neck"
"head"
"leftshoulder"
"leftarm"
"leftforearm"
"lefthand"
"rightshoulder"
"rightarm"
"rightforearm"
"righthand"

Return Values

None.

Description

Attach the entity to the specified bone of its parent. In addition to specifying the bone to which the entity is to be attached, it is also necessary to specify the person to which the bone belongs. This is done by calling Entity.AttachToParent and supplying the desired person.

Examples

```
entity = Entity.Create()
Entity.AttachToParent( entity, LocalPlayer.GetPerson() )
Entity.AttachToBone( entity, "head" )
```

See Also

- Entity.Create
- Entity.AttachToParent
- Entity.SetAttachType

Entity.AttachToParent

Brief

Attaches the entity to the specified parent.

Definition

```
Entity.AttachToParent( Entity child, Entity parent, string boneName = nil )
Entity.AttachToParent( Entity child, Person parent, string boneName = nil )
```

Arguments

entity - The entity to update.parent - The parent to attach the entity to or nil to remove it.boneName - The name of the bone to attach to, if any.

For a person, it must be from the following list:
"pelvis"
"hips"
"leftleg"
"leftfoot"
"rightleg"
"rightfoot"
"spine"
"neck"
"head"
"leftshoulder"
"leftarm"
"leftforearm"
"lefthand"
"rightshoulder"
"rightarm"
"rightforearm"
"righthand"

Return Values

None.

Description

Attaches the entity to the specified parent. The parent's world matrix will be applied to the entity so the entity's position, rotation and scale are relative to the parent. If a bone is specified, the entity will be attached to the specified bone in the parent's skeleton.

Examples

```
-- Parent an entity to an entity
entity1 = Entity.Create()
parent = Entity.Create()
Entity.AttachToParent( entity1, parent );

-- Parent an entity to a person's right hand
entity2 = Entity.Create()
person = Person.FindInInstance( personId )
Entity.AttachToParent( entity2, person, "righthand" )
```

See Also

- Entity.Create
- Person.FindInInstance
- Entity.SetAttachType

Entity.BlendAnimIn

Brief

Blends an animation into, and alongside the currently playing animations.

Definition

```
Entity.BlendAnimIn( Entity entity, string resourceName, boolean isLooping = true, number time = 0,
number easeType = AnimBlendType.Linear, number blendDuration = 0.25, number priority = 0, Animation
anim = nil)
Entity.BlendAnimIn( Entity entity, Resource animResource, boolean isLooping = true, number time = 0,
number easeType = AnimBlendType.Linear, number blendDuration = 0.25, number priority = 0, Animation
anim = nil)
```

Arguments

entity - entity to blend animation ontoresourceName - the name of the animation resource to blend inanimResource - the animation resource to blend inisLooping - flag indicating if the animation played should loop. (default is true)time - point in at which the animation should start playing from (default is 0.0)easeType - the easing method to be used. The values used are in the enumerated type AnimBlendType (default value is AnimBlendType.Linear)blendDuration - the period in seconds, over which the blend factor will increase from 0.0 to 1.0 (default is 0.25)priority - The priority affects where in the animation list an animation will be played. The higher the number, the later it will be played.
If a prior animation is being played at the same priority, then the new animation will be placed after the old animation in the list. (default is 0)anim - if supplied, the animation handle is set to the newly created active animation

Return Values

None.

Description

Values for the ease type are provided by the enumeration AnimBlendType:
AnimBlendType.None
AnimBlendType.Linear
AnimBlendType.SCurve
AnimBlendType.Accelerate
AnimBlendType.Decelerate

Examples

```
Entity.BlendAnimIn( entity, "myAnim" )
Entity.BlendAnimIn( entity, "myWalk", true, 0.0, AnimBlendType.Linear, 0.25, 0, animHandleToSave)
```

See Also

- Entity.GetIndexOfAnim
- Entity.GetActiveAnim

Entity.BlendAnimOut

Brief

Blends the specified active animation out.

Definition

```
Entity.BlendAnimOut(Entity entity, Animation anim, number easeType = AnimBlendType.Linear, number
blendDuration = 0.25)
```

Arguments

entity - entity to blend anim out fromanim - animation to blend outeaseType - the blend curve to use when blending out, default is AnimBlendType.LinearblendDuration - the duration over which the animation is blended out, default is 0.25

Return Values

None.

Description

Begins to blend out a currently active animation. To check the progress of the blend, use Animation.GetBlendState and/or Animation.GetBlendTime. An invalid animation handle will have no effect.

Examples

```
local animIndex = Entity.GetIndexOfAnim( entity, "idleAnim" )
local hAnim = Entity.GetActiveAnim( entity, animIndex )
Entity.BlendAnimOut( entity, hAnim, AnimBlendType.Linear, 0.25 )
```

See Also

- Entity.BlendAnimIn
- Entity.GetIndexOfAnim
- Entity.GetActiveAnim
- Entity.PlayAnim
- Animation.GetBlendWeight

Entity.Copy

Brief

Clone an entity's state to another entity.

Definition

```
Entity.Copy(Entity dest, Entity source)
```

Arguments

dest - Destination entity to modify.source - Source entity to copy.

Return Values

None.

Description

Copies all state from the source entity to the destination entity, leaving the source entity unchanged. The only state that will not be copied is transform state (position and orientation), as having two identical entities in the same place may cause issues with collision and other systems. The destination entity is still a different entity (it will not compare equal and will have a different unique ID) but will have the same state as the source entity. Similarly to Reset, the hierarchy state of the destination entity will not be modified; if the script wishes to detach the entity and its children this must be done explicitly.

Examples

```
myDestEntity:Copy(mySourceEntity)
```

See Also

- Entity.Reset

Entity.Create

Brief

Creates a new entity.

Definition

```
Entity Entity.Create()
```

Arguments

None

Return Values

A new Entity instance.

Description

Creates a new entity.

Examples

```
entity = Entity.Create()
```

See Also

None

Entity.CreateForceAction

Brief

Creates a new 'force' action for an entity.

Definition

```
number Entity.CreateForceAction( Entity entity, boolean autoEnable = false )
```

Arguments

entity - The entity to which the action will belong.autoEnable - true to automatically enable the action, false otherwise.

Return Values

A numerical action ID.

Description

Creates a new 'force' action for an entity. If autoEnable is false, the action can be enabled with Entity.EnableAction. This action applies a force to an entity every physics update. This function is only available if the entity has a collision item.

Examples

```
entity = Entity.Create()
e:SetCollision("kCollisionResource")
e:EnableDynamics()
actionId = Entity.CreateForceAction( e, false )
f = Vector4.Create( 10.0, 0.0, 0.0 )
Entity.EnableAction( e, actionId )
Entity.SetActionForce( e, actionId, f )
```

See Also

- Entity.CreateTorqueAction
- Entity.CreatePointForceAction
- Entity.EnableAction
- Entity.SetActionForce

Entity.CreatePointForceAction

Brief

Creates a new 'point-force' action for an entity.

Definition

```
number Entity.CreatePointForceAction( Entity entity, boolean localSpace = true, boolean autoEnable =
false )
```

Arguments

entity - The entity to which the action will belonglocalSpace - true to use a local-space point, false to use a world-space point.autoEnable - true to automatically enable the action, false otherwise.

Return Values

A numerical action ID.

Description

Creates a new 'point-force' action for an entity. If autoEnable is false, the action can be enabled with Entity.EnableAction. This action applies a force at a given position to an entity. The position can a local-space point or a world-space point. This function is only available if the entity has a collision item.

Examples

```
entity = Entity.Create()
e:SetCollision("kCollisionResource")
e:EnableDynamics()
actionId = Entity.CreatePointForceAction( e, true, false )
f = Vector4.Create( 10.0, 0.0, 0.0 )
p = Vector4.Create( 0.0, 1.0, 0.0 )
Entity.EnableAction( e, actionId )
Entity.SetActionForce( e, actionId, f )
Entity.SetActionPosition( e, actionId, p )
```

See Also

- Entity.CreateTorqueAction
- Entity.CreateForceAction
- Entity.SetActionPosition

Entity.CreateTorqueAction

Brief

Creates a new 'torque' action for an entity.

Definition

```
number Entity.CreateTorqueAction( Entity entity, boolean autoEnable = false )
```

Arguments

entity - The entity to which the action will belong.autoEnable - true to automatically enable the action, false otherwise.

Return Values

A numerical action ID.

Description

Creates a new 'torque' action for an entity. If autoEnable is false, the action can be enabled with Entity.EnableAction. This action applies a torque to an entity every physics update. This function is only available if the entity has a collision item.

Examples

```
entity = Entity.Create()
e:SetCollision("kCollisionResource")
e:EnableDynamics()
actionId = Entity.CreateTorqueAction( e, false )
t = Vector4.Create( 10.0, 0.0, 0.0 )
Entity.EnableAction( e, actionId )
Entity.SetActionTorque( e, actionId, t )
```

See Also

- Entity.CreateForceAction
- Entity.CreatePointForceAction
- Entity.EnableAction
- Entity.SetActionTorque

Entity.DebugSetAxesVisible

Brief

Sets the entity to show its axis or not.

Definition

```
Entity.DebugSetAxesVisible( Entity entity, boolean value, boolean recursive, number size = 1)
```

Arguments

entity - The entity to update.value - If true the entity shows its axis and if false it hides it.recursive - If true the entity also sets the values of all the children below it.size - The size of the line drawn to represent the axis (in meters).

Return Values

None.

Description

Sets the entity to show its axis or not. Entities default to not show their axis when created. Note that this function is provided for debugging purposes and should not be used in a finalized script.

Examples

```
entity = Entity.Create()
Entity.DebugSetAxesVisible( entity, true, true, 1.0 )
```

See Also

- Entity.DebugSetHierarchyVisible
- Entity.DebugSetCollisionVisible

Entity.DebugSetCollisionVisible

Brief

Sets the entity to show its collision or not.

Definition

```
Entity.DebugSetCollisionVisible( Entity entity, boolean value, boolean recursive )
```

Arguments

entity - The entity to update.value - If true the entity shows its collision shape.recursive - If true the entity also sets the values of all the children below it.

Return Values

None.

Description

Sets the entity to show its collision or not. Entities default to not show their collision when created. Note that this function is provided for debugging purposes and should not be used in a finalized script.

Examples

```
entity = Entity.Create()
Entity.DebugSetCollisionVisible( entity, true, true )
```

See Also

- Entity.DebugSetAxesVisible
- Entity.DebugSetHierarchyVisible

Entity.DebugSetHierarchyVisible

Brief

Sets the entity to show its hierarchy or not.

Definition

```
Entity.DebugSetHierarchyVisible( Entity entity, boolean value, boolean recursive )
```

Arguments

entity - The entity to update.value - If true the entity shows its hierarchy with its immediate children and if false it hides it.recursive - If true the entity also sets the values of all the children below it.

Return Values

None.

Description

Sets the entity to show its hierarchy or not. Entities default to not show their hierarchy when created. Note that this function is provided for debugging purposes and should not be used in a finalized script.

Examples

```
entity = Entity.Create()
Entity.DebugSetHierarchyVisible( entity, true, true )
```

See Also

- Entity.DebugSetAxesVisible
- Entity.DebugSetCollisionVisible

Entity.DestroyAction

Brief

Destroy an entity action.

Definition

```
Entity.DestroyAction( Entity entity, number actionId )
```

Arguments

entity - The entity to which the action belongsactionId - The numerical action ID, obtained at action creation time

Return Values

None.

Description

Destroy an entity action. It is good practice to destroy all actions when no longer required, however the system will also automatically destroy them if the target entity is destroyed. This function is only available if the entity has a collision item.

Examples

```
entity = Entity.Create()
e:SetCollision("kCollisionResource")
e:EnableDynamics()
actionId = Entity.CreatePointForceAction( e, false )
f = Vector4.Create( 10.0, 0.0, 0.0 )
p = Vector4.Create( 0.0, 1.0, 0.0 )
Entity.EnableAction( e, actionId )
Entity.SetActionForce( e, actionId, f )
Entity.SetActionPosition( e, actionId, p )
...
Entity.DestroyAction( e, actionId )
```

See Also

- Entity.CreateForceAction
- Entity.CreateTorqueAction

Entity.DetachPersonLabel

Brief

Detaches a previously attached person label from the entity.

Definition

```
Entity.DetachPersonLabel(Entity entity, Person person)
```

Arguments

person - The person the label was taken from.

Return Values

Description

This function will detach the person label previously attached to the entity, and will re-attach it to the person it was taken from.

Examples

```
Entity.DetachPersonLabel(entity, person)
```

See Also

- Entity.AttachPersonLabel

Entity.EnableAction

Brief

Make entity action take effect.

Definition

```
Entity.EnableAction( Entity entity, number actionId, boolean enable = true )
```

Arguments

entity - The entity to which the action belongsactionId - The numerical action IDenable - true to enable the action, false otherwise

Return Values

None.

Description

Make entity action take effect. A disabled action will not be applied to the entity. This function is only available if the entity has a collision item.

Examples

```
entity = Entity.Create()
e:SetCollision("kCollisionResource")
e:EnableDynamics()
actionId = Entity.CreatePointForceAction( e, false )
f = Vector4.Create( 10.0, 0.0, 0.0 )
p = Vector4.Create( 0.0, 1.0, 0.0 )
Entity.EnableAction( e, actionId, true )
Entity.SetActionForce( e, actionId, f )
Entity.SetActionPosition( e, actionId, p )
...
Entity.DestroyAction( e, actionId )
```

See Also

- Entity.IsActionEnabled

Entity.EnableCollision

Brief

Enables/disables collision on the specified entity.

Definition

```
Entity.EnableCollision( Entity entity, boolean enable = true, Phys3dWorld world = nil )
```

Arguments

entity - The entity to update.enable - true to enable collision, false otherwise (defaults to true if not specified).world - The Phys3dWorld to add to the entity to, or nil to add to the default world.

Return Values

None.

Description

If collision is enabled for the specified entity, the local player's avatar and other physical entities can collide with it, thereby preventing penetrations. If collision is disabled, the local player's avatar and other physical entities will be able to pass through this entity. Note that the collision world specified in this call will override the collision world specified to SetCollision, if any; specifying nil here will cause the item to be removed from the world specified in SetCollision and added to the default Phys3dWorld. Limitation: Although collision can be enabled for translated and/or rotated entities, scaling is not supported.

Examples

```
-- Turn off collisions
Entity.EnableCollision( entity, false )

-- Turn on collisions
Entity.EnableCollision( entity )
```

See Also

- Entity.EnableDynamics
- Phys3dWorld.Create

Entity.EnableCollisionCallbacks

Brief

Enable or disable collision callbacks for the specified entity.

Definition

```
Entity.EnableCollisionCallbacks(Entity entity, boolean enable)
```

Arguments

entity - the entity to enable/disable collision callbacks forenable - enable or disable

Return Values

None.

Description

Enable or disable collision callbacks for the specified entity. Note that in order to receive callbacks, you must register your interest in the particular type(s) of collision event you want to receive, using Entity.RegisterCollisionCallback(). You must also set the maximum contacts using Entity.SetMaxContacts(). Finally, the entity must have collision set and enabled for the callback(s) to occur. The entity need not have dynamics enabled as static entities will also report collisions.

Examples

```
-- set up a collision data buffer
Entity.SetMaxContacts( 64 )

foo = Entity.Create()
foo:SetModel("model")
foo:SetCollision("collision")
foo:SetPosition(Vector4.Create(0, 5, 0))
foo:SetVisible(true)
foo:EnableCollision(true)
foo:EnableDynamics(true)
foo:EnableCollisionCallbacks(true)
foo:RegisterCollisionCallback(EntityCollision.ContactAdded, ContactAddedCallback, foo)
```

See Also

- Entity.RegisterCollisionCallback
- Entity.SetMaxContacts
- Entity.SetCollision
- Entity.EnableCollision

Entity.EnableDynamics

Brief

Enables/disables dynamics on the specified entity.

Definition

```
Entity.EnableDynamics( Entity entity, boolean enable = true )
```

Arguments

entity - The entity to update.enable - true to enable dynamics, false otherwise (defaults to true if not specified).

Return Values

None.

Description

If dynamics is enabled, the rigid body attached to the entity will be activated in PS Home's physics simulation. If dynamics is disabled, the entity will become fixed in its current position and will no longer be affected by the simulation. Note that collision must be enabled on the entity if dynamics is to be enabled. Attempts to set absolute translations or rotations on an entity that has dynamics enabled will be ignored. For this to succeed, dynamics must be temporarily disabled while the transformation is explicitly applied to the entity. Limitation: This function must not be called for entities that have been scaled in any way.

Examples

```
-- Turn off dynamics
Entity.EnableDynamics( entity, false )

-- Turn on dynamics (collision must be enabled as well)
Entity.EnableCollision( entity )
Entity.EnableDynamics( entity )
```

See Also

- Entity.EnableCollision

Entity.EnableDynamicsAutoDeactivation

Brief

Enable/disable auto-deactivation of dynamics on the specified entity.

Definition

```
Entity.EnableDynamicsAutoDeactivation( Entity entity, boolean enable = true )
```

Arguments

entity - The entity to update.enable - true to enable auto-deactivation, false otherwise (defaults to true if not specified).

Return Values

None.

Description

Dynamics must be enabled with Entity.EnableDynamics() for the activated/deactivated state to take effect. An Entity with activated dynamics is one that is having it's motion updated. If dynamics are deactivated, Home's physics engine will stop updating the Entity's motion, until it is activated again. Entities can be activated by another call to Entity.SetDynamicsActive or by the application of forces/torques/impulses or by coming into contact with other activated Entities. Entities can be deactivated with Entity.SetDynamicsActive or by allowing auto-deactivation which will deactivate the Entity if it has not been moving for a while. Auto-deactivation can be enabled/disabled with this function.

Examples

```
-- Disable auto-deactivation
Entity.EnableDynamicsAutoDeactivation( entity, false )

-- Enable auto-deactivation
Entity.EnableCollision( entity )
Entity.EnableDynamics( entity )
Entity.SetDynamicsActive( entity )
Entity.EnableDynamicsAutoDeactivation( entity )
```

See Also

- Entity.EnableDynamics
- Entity.SetDynamicsActive
- Entity.IsDynamicsAutoDeactivationEnabled

Entity.EnableInterpolatedMode

Brief

Enable/disable interpolated mode.

Definition

```
Entity.EnableInterpolatedMode( Entity entity, boolean enable )
```

Arguments

entity - The entity.enable - true to enable interpolated mode, false otherwise.

Return Values

None

Description

Enable/disable interpolated mode. Interpolated mode can help smooth the motion of dynamic entities, however there is an extra processing overhead.

Examples

```
Entity.EnableInterpolatedMode( entity, true )
```

See Also

- Entity.IsInterpolatedModeEnabled

Entity.EnableKeepUpright

Brief

Set whether or not to keep the entity upright.

Definition

```
Entity.EnableKeepUpright( Entity entity, boolean keepUpright = true, number softness = 0.0 )
```

Arguments

entity - entity to be affected.keepUpright - true to keep the entity upright, false otherwisesoftness - the softness of the constraint: 0 is strongest, 1 is weakest.

Return Values

None.

Description

Adds a dynamic constraint to try to keep the entity upright. The softness value is only necessary when enabling the keep upright constraint.

Examples

```
e = Entity.Create()
e:SetCollision("kCollisionResource")
e:EnableCollision()
e:EnableDynamics()
e:EnableKeepUpright( true, 0.1 )
```

See Also

- Entity.IsKeepUprightEnabled

Entity.EnableLayerCollision

Brief

Set whether or not an entity can collide with collision of the specified layer.

Definition

```
Entity.EnableLayerCollision( Entity entity, enum layer, boolean enable )
```

Arguments

entity - The entity.layer - The collision layer (a CollisionLayer enum).enable - true to enabled collision, false otherwise.

Return Values

None.

Description

Set whether or not an entity can collide with collision of the specified layer. Changes will only take effect for new collisions/overlaps. If two objects are already colliding (or very close), disabling their collisions via this function will not take effect until they separate. Also, if two objects are overlapping but do not currently collide with each other (due to disabled layer collisions), then enabling their collision via this function will not take effect until they separate.

Examples

```
Entity.EnableLayerCollision( entity1, Entity.GetCollisionLayer( entity2 ), false )
```

See Also

- Entity.IsLayerCollisionEnabled

Entity.EvaluateAnims

Brief

Evaluates all current animation(s) active on the entity, and updates the skeleton.

Definition

```
Entity.EvaluateAnims( Entity entity )
```

Arguments

entity - The entity to update.

Return Values

None.

Description

In the general case, changes made to the animation state on an entity will be reflected in the next frame. In certain situations, a script may wish to modify the animation list and have it immediately reflected in the current frame. In this event, the script can call this function, and PS Home will re-evaluate the entire animation blend list on the entity, and apply the results to the entity's skeleton. Use this function sparingly, as this evaluation is synchronous and significantly slower than the core animation update. Generally the single-frame delay in the normal case can be hidden with appropriate blending, rather than relying on this function.

Examples

```
entity = Entity.Create()
Entity.SetModel( entity, "mymodel" )
Entity.SetSkeleton( entity, "myskel" )
Entity.PlayAnim( entity, "myanim" )
...
function OnUpdate()
    if (playInstantDeathAnim) then
        local animIndex = Entity.GetIndexOfAnim( entity, "myanim" )
        local anim = Entity.GetActiveAnim( entity, animIndex )
        Entity.BlendAnimOut( entity, anim, AnimBlendType.Linear, 0.1 )
        Entity.BlendAnimIn( entity, "instantDeath", false, 0.0, AnimBlendType.Linear, 0.1, 0)
        Entity.EvaluateAnims( entity )
    end
end
```

See Also

None

Entity.FindMaterials

Brief

Get an array of materials on an entity's model.

Definition

```
Material[] Entity.FindMaterials( Entity entity, string materialId=nil)
```

Arguments

entity - The entity to search on.materialId - The material's scriptId as set in max/maya, or nil to get all materials on the entity

Return Values

An array of Materials that matches the given materialId, or all materials on the entity if no id is given.

Description

Gets an array of Materials that matches the given materialId, or all materials on the entity if no id is given.

Examples

```
entity = Entity.Create()
Entity.SetModel( entity, "mymodel" )
allMaterials = Entity.FindMaterials(entity)
print( "Found " .. #allMaterials .. " materials." );
for i, v in ipairs(array) do
    print( tostring( v ) )
end
```

See Also

- Material.GetParam
- Material.SetAlphaRef
- Material.SetBlendMode
- Material.SetMatrix
- Material.SetTexture
- Material.SetVector

Entity.GetActiveAnim

Brief

Returns a handle to a currently active animation

Definition

```
Animation Entity.GetActiveAnim( Entity entity, number index )
Entity.GetActiveAnim( Entity entity, number index, Animation anim )
```

Arguments

entity - the entity to accessindex - the index of the animation handle (within the entity's animation list) to retrieve.anim - the animation object to set, instead of returning a new one

Return Values

If animation is not specified as a parameter, either a handle to the animation, or nil if no current anim.
If an animation is specified as a parameter, there is no return value.

Description

Get a handle to the current active animation. Note that if there is no current active animation and the user has specified an animation object to set, the animation object will be marked as invalid. To check this, use Animation.IsValid.

Examples

```
local animIndex = entity:GetIndexOfAnim("myAnim")
local anim = entity:GetActiveAnim( animIndex )
```

See Also

- Animation.IsValid
- Animation.Create

Entity.GetAngularDamping

Brief

Gets the angular damping of the collision resource attached to the entity.

Definition

```
number Entity.GetAngularDamping( Entity entity )
```

Arguments

entity - The entity to query.

Return Values

The angular damping for the entity.

Description

Gets the angular damping of the collision resource attached to the entity. This function is only valid if the entity has a collision item.

Examples

```
local damping = Entity.GetAngularDamping( entity )
```

See Also

- Entity.EnableCollision
- Entity.EnableDynamics
- Entity.SetCollision
- Entity.SetAngularDamping

Entity.GetAngularVelocity

Brief

Returns the angular velocity around the center of mass of the entity.

Definition

```
Vector4 Entity.GetAngularVelocity( Entity entity )
Entity.GetAngularVelocity( Entity entity, Vector4 outVec )
```

Arguments

entity - The entity to query.outVec - The Vector4 to store the result to.

Return Values

The entity's current angular velocity.

Description

Returns the angular velocity around the center of mass of the entity's rigid body in world space. The W component of the returned vector is zero. If outVec is specified it stores the result in there instead of returning a new Vector4.

Examples

```
-- Add the entity to the physics simulation
Entity.EnableCollision( entity, true )
Entity.EnableDynamics( entity, true )

-- Query the angular velocity
angularVel = Entity.GetAngularVelocity( entity )
```

See Also

- Entity.GetLinearVelocity

Entity.GetBoneCount

Brief

Get the number of bones in the entity's skeleton.

Definition

```
number Entity.GetBoneCount(Entity entity)
```

Arguments

entity - Entity to query.

Return Values

Number of bones in the skeleton, or 0 if the entity has no skeleton attached.

Description

Retrieves the number of bones/joints in the entity's skeleton. If no skeleton is present, 0 is returned.

Examples

```
local boneCount = myEntity:GetBoneCount()
```

See Also

- Entity.GetBoneIndex

Entity.GetBoneIndex

Brief

Find the index of a bone in the entity's skeleton.

Definition

```
number Entity.GetBoneIndex(Entity entity, string boneName)
```

Arguments

entity - Entity to query.boneName - Name of bone to locate.

Return Values

The index of the bone in the skeleton (in range 1 - GetBoneCount()), or 0 if not found.

Description

Retrieves the index of the specified bone/joint in the entity's skeleton. The index is 1-based. If the bone cannot be found, or there is no skeleton attached, 0 is returned.

Examples

```
local leftHandIndex = myEntity:GetBoneIndex("lefthand")
```

See Also

- Entity.GetBoneCount
- Entity.GetBoneLocalMatrix
- Entity.GetBoneWorldMatrix

Entity.GetBoneLocalMatrix

Brief

Get the local transform matrix for a bone in the skeleton.

Definition

```
Entity.GetBoneLocalMatrix(Entity entity, string|number bone, Matrix44 result)
```

Arguments

entity - Entity to query.bone - Bone name or index.result (out) - Matrix44 value to store result into.

Return Values

None.

Description

Retrieve the local-space transform matrix for a specified bone in the skeleton. If the bone is not found an error will result. Retrieving the matrix by index will be more efficient if done more than once.

Examples

```
local mat = Matrix44.Create()
myEntity:GetBoneLocalMatrix("lefthand", mat)
-- or
local leftHandIndex = myEntity:GetBoneIndex("lefthand")
myEntity:GetBoneLocalMatrix(leftHandIndex, mat)
```

See Also

- Entity.GetBoneWorldMatrix
- Entity.GetBoneIndex

Entity.GetBoneWorldMatrix

Brief

Get the world transform matrix for a bone in the skeleton.

Definition

```
Entity.GetBoneWorldMatrix(Entity entity, string|number bone, Matrix44 result)
```

Arguments

entity - Entity to query.bone - Bone name or index.result (out) - Matrix44 value to store result into.

Return Values

None.

Description

Retrieve the world-space transform matrix for a specified bone in the skeleton. If the bone is not found an error will result. Retrieving the matrix by index will be more efficient if done more than once.

Examples

```
local mat = Matrix44.Create()
myEntity:GetBoneWorldMatrix("lefthand", mat)
-- or
local leftHandIndex = myEntity:GetBoneIndex("lefthand")
myEntity:GetBoneWorldMatrix(leftHandIndex, mat)
```

See Also

- Entity.GetBoneLocalMatrix
- Entity.GetBoneIndex

Entity.GetCenterOfMassLocal

Brief

Returns the position of the entity's center of mass in local space.

Definition

```
Vector4 Entity.GetCenterOfMassLocal( Entity entity )
Entity.GetCenterOfMassLocal( Entity entity, Vector4 outVec )
```

Arguments

entity - The entity to query.outVec - The Vector4 to store the result to.

Return Values

The position of the entity's center of mass in local space.

Description

Returns the position of the entity's center of mass in local space. This may be offset from the coordinate origin of the entity depending on its

physics configuration. If outVec is specified it is used to store the result instead of returning a new Vector4. This function is only available if the entity has a collision item.

Examples

```
print( "Entity COM = " .. Entity.GetCenterOfMassLocal( entity ) )
```

See Also

- Entity.EnableDynamics
- Entity.GetCenterOfMassLocal

Entity.GetCenterOfMassWorld

Brief

Returns the position of the entity's center of mass in world space.

Definition

```
Vector4 Entity.GetCenterOfMassWorld( Entity entity )
Entity.GetCenterOfMassWorld( Entity entity, Vector4 outVec )
```

Arguments

entity - The entity to query.outVec - The Vector4 to store the result to.

Return Values

The position of the entity's center of mass in world space.

Description

Returns the position of the entity's center of mass in world space. This may differ from the position of the entity depending on its physics configuration. If outVec is specified it stores the result in there instead of returning a new Vector4.

Examples

```
print( "Entity COM = " .. Entity.GetCenterOfMassWorld( entity ) )
```

See Also

- Entity.EnableDynamics

Entity.GetChildren

Brief

Get an array of the child entities of a specified entity.

Definition

```
table Entity.GetChildren(Entity entity)
```

Arguments

entity - entity to obtain children from.

Return Values

An array of the child entities of the specified entity.

Description

If an entity has had children attached using Entity.AttachToParent, this function will return an array of all the child entities, in the order in which they were attached. If there are no children, an empty table will be returned.

Examples

```
parent = Entity.Create()
child1 = Entity.Create()
child2 = Entity.Create()
child1:AttachToParent(parent)
child2:AttachToParent(parent)
children = parent:GetChildren()
print(#children)                -- prints 2, children[1] == child1 and children[2] == child2
print(#child1:GetChildren())    -- prints 0, child1 has no children
```

See Also

- Entity.AttachToParent
- Entity.GetParent
- Entity.GetSiblings

Entity.GetCloth

Brief

Retrieve a handle to the entity's cloth meshes, if any.

Definition

```
Cloth Entity.GetCloth(Entity entity)
```

Arguments

entity - Entity to query.

Return Values

A Cloth object, or nil if the entity has no model or model contains no cloth elements.

Description

If the entity's model contains cloth elements, this function can be used to obtain a handle to the cloth elements to alter their parameters. All cloth elements in the model are modified when using the cloth handle.

Examples

```
myEntity:SetModel("ModelWithCloth")
local cloth = myEntity:GetCloth()
if (cloth) then
    cloth:SetWindAngle(135)
end
```

See Also

- Cloth.SetWindAngle

Entity.GetCollisionLayer

Brief

Query the collision layer of an entity.

Definition

```
enum Entity.GetCollisionLayer( Entity entity )
```

Arguments

entity - The entity.

Return Values

The collision layer (a CollisionLayer enum).

Description

Query the collision layer of an entity.

Examples

```
print( "collisionLayer="..Entity.GetCollisionLayer( entity )
```

See Also

- Entity.SetCollisionLayer

Entity.GetContactCount

Brief

Retrieve the number of contacts passed to a collision callback.

Definition

```
number Entity.GetContactCount()
```

Arguments

None

Return Values

Number of contacts passed to collision callback.

Description

Retrieve the number of contacts available in a collision callback. This function is only valid inside a collision callback.

Examples

```
function ContactAddedCallback(entity)
    local numContacts = Entity.GetContactCount()
    print(numContacts .. ' contacts added')
end
```

See Also

- Entity.GetContactPoint
- Entity.GetContactEntity
- Entity.GetContactRelVelocity
- Entity.GetContactId

Entity.GetContactEntity

Brief

Retrieve the other entity that has been contacted in a collision.

Definition

```
Entity Entity.GetContactEntity(number index)
```

Arguments

index - index of the contact point

Return Values

The contacted entity, or nil in a collision with the world.

Description

Retrieve the other entity contacted in a collision, that is, the entity that is not reporting the collision. In a collision with world geometry or another physics object not owned by an Entity, nil is returned. It is important to realise that the entity returned is not the exact same entity instance from Lua's perspective. It will compare equal using == or ~=, and all API functions are available, but if used as a table key it will not match the "original" entity. If you wish to match the entities using a table, it is recommended to insert them using a unique key based on the entity itself. Note that this function is only valid from within a collision callback, and will not be available for ContactRemoved callbacks.

Examples

```
function ContactAddedCallback(entity)
    local numContacts = Entity.GetContactCount()
    for i = 1, numContacts do
        local entity = Entity.GetContactEntity(i)
        if (entity == nil) then
            print("I've hit the world!\n")
        end
    end
end
```

See Also

- Entity.GetContactCount
- Entity.GetContactPoint
- Entity.GetContactRelVelocity

- Entity.GetContactId

Entity.GetContactId

Brief

Retrieve the unique contact ID.

Definition

```
number Entity.GetContactId(number index)
```

Arguments

index - index of the contact point

Return Values

The contact ID.

Description

Retrieve the unique ID number assigned the contact. This number will be preserved across multiple frames and so can be used to track individual contacts over the duration of the collision. Do not make any assumptions about the ID numbers returned. Note that this function is only valid from within a collision callback.

Examples

```
function ContactProcessCallback(entity)
    local numContacts = Entity.GetContactCount()
    for i = 1, numContacts do
        local id = Entity.GetContactId(i)
        UpdateContactTracker(id)                -- do something with contacts being tracked
    end
end
```

See Also

Entity.GetContactPoint

Brief

Retrieve the position and/or normal of the specified contact.

Definition

```
Entity.GetContactPoint(number index, Vector4 position = nil, Vector4 normal = nil)
```

Arguments

index - index of the contact pointposition - Vector4 to store the position into, or nil if unwantednormal - Vector4 to store the normal into, or nil if unwanted

Return Values

None.

Description

Retrieve the position and/or normal of a contact, specified by an index which must be between 1 and the contact count. Passing nil to position and/or normal will skip retrieval of that value. Both values will be given in world space. Note that this function is only valid from within a collision callback, and will not be available for ContactRemoved callbacks.

Examples

```
function ContactAddedCallback(entity)
    local numContacts = Entity.GetContactCount()
    local pos = Vector4.Create()
    local norm = Vector4.Create()
    for i = 1, numContacts do
        Entity.GetContactPoint(i, pos, norm)
        SpawnImpactEffect(pos, norm)            -- user function to spawn an impact particle effect
    end
end
```

See Also

- Entity.GetContactCount
- Entity.GetContactEntity
- Entity.GetContactRelVelocity
- Entity.GetContactId

Entity.GetContactRelVelocity

Brief

Retrieve the relative velocity of the colliding objects.

Definition

```
number Entity.GetContactRelVelocity(number index)
```

Arguments

index - index of the contact point

Return Values

The relative velocity, projected onto the contact normal.

Description

Retrieve the relative velocity of the colliding objects, projected onto the contact normal. Note that the value may be negative or positive. Note also that this function is only valid from within a collision callback, and only in reponse to the ContactAdded event.

Examples

```
function ContactAddedCallback(entity)
    local numContacts = Entity.GetContactCount()
    local pos = Vector4.Create()
    local norm = Vector4.Create()
    for i = 1, numContacts do
        local relVel = math.abs(Entity.GetContactRelVelocity(i))
        Entity.GetContactPoint(i, pos, norm)
        if (relVel > 10) then
            SpawnBigParticle(pos, norm)
        else
            SpawnSmallParticle(pos, norm)
        end
    end
end
```

See Also

Entity.GetContactUserId

Brief

Retrieve the user ID of the part of a party (entity or collision geometry) involved in a contact.

Definition

```
number, number Entity.GetContactUserId(number contactIndex)
number Entity.GetContactUserId(number contactIndex, number partyIndex)
```

Arguments

contactIndex - index of the contact pointpartyIndex - index of the party; 1 or 2

Return Values

In the first usage, both user IDs in order, or the single user ID requested in the second usage. If a value is nil, no user ID is available.

Description

Retrieve the user ID(s) of the part of a party (entity or collision geometry) involved in a contact. Note that this function is only valid from within a collision callback. Note that there are two collision items involved in a contact, so there are two corresponding user IDs. The user IDs are ordered such that party 1 is the entity registered for collision callbacks, and party 2 is the other collision item.

Examples

```
function ContactProcessCallback(entity)
    local numContacts = Entity.GetContactCount()
    for i = 1, numContacts do
        local idA = Entity.GetContactUserId(i, 1)
        local idB = Entity.GetContactUserId(i, 2)
        -- or local idA, idB = Entity.GetContactUserId(i)
        if idA == 100 then
            print("UserID 100 has been hit!")
        end
    end
end
```

See Also

Entity.GetCullDistance

Brief

Get the cull distance for an entity.

Definition

```
number Entity.GetCullDistance(Entity entity)
```

Arguments

entity - The entity to modify.

Return Values

The cull distance of the entity, or 0 if disabled.

Description

Get the cull distance of an entity set with SetCullDistance.

Examples

```
if (myEntity:GetCullDistance() > 500) then
    print("This entity has a very large cull distance!")
end
```

See Also

- Entity.SetCullDistance

Entity.GetGravity

Brief

Query the gravity vector of the entity.

Definition

```
Vector4 Entity.GetGravity( Entity entity )
Entity.GetGravity( Entity entity, Vector4 gravityVectorOut )
```

Arguments

entity - entity to be queried.gravityVectorOut - gravity vector storage.

Return Values

The gravity vector

Description

Query the gravity vector of the entity. This function is only valid if the entity has a collision item.

Examples

```
-- first form
print( Entity.GetGravity( entity ) )

-- second form
local   grav = Vector4.Create()
Entity.GetGravity( entity, grav )
print( grav )
```

See Also

- Entity.SetGravity

Entity.GetIndexOfAnim

Brief

Find the index of the active animation that is using the specified animation resource.

Definition

```
number Entity.GetIndexOfAnim( Entity entity, string resourceName, number startIndex = 1 )
```

Arguments

entity - the entity to queryresourceName - the name of the animation we are looking forstartIndex - index to begin search at

Return Values

The index of the animation being queried, or nil if not found.

Description

The start index is provided so that multiple searches can be made if the same resorce is being played multiple times.

Examples

```
local animIndex = Entity.GetIndexOfAnim( myEntity, "myAnim" )
```

See Also

- Entity.GetActiveAnim
- Entity.BlendAnimIn
- Entity.BlendAnimOut
- Entity.StopAnim

Entity.GetLinearDamping

Brief

Gets the linear damping of the collision resource attached to the entity.

Definition

```
number Entity.GetLinearDamping( Entity entity )
```

Arguments

entity - The entity to query.

Return Values

The linear damping for the entity.

Description

Gets the linear damping of the collision resource attached to the entity. This function is only valid if the entity has a collision item.

Examples

```
local damping = Entity.GetLinearDamping( entity )
```

See Also

- Entity.EnableCollision
- Entity.EnableDynamics
- Entity.SetCollision
- Entity.SetLinearDamping

Entity.GetLinearVelocity

Brief

Returns the linear velocity of the entity.

Definition

```
Vector4 Entity.GetLinearVelocity( Entity entity )
Entity.GetLinearVelocity( Entity entity, Vector4 outVec )
```

Arguments

entity - The entity to query.outVec - The vector to write the result to.

Return Values

The entity's current linear velocity.

Description

Returns the linear velocity of the center of mass of the entity's rigid body in world space. The W component of the returned vector is zero. If outVec is specified it stores the result in there instead of returning a new Vector4.

Examples

```
-- Add the entity to the physics simulation
Entity.EnableCollision( entity, true )
Entity.EnableDynamics( entity, true )

-- Query the linear velocity
linearVel = Entity.GetLinearVelocity( entity )
```

See Also

- Entity.GetAngularVelocity

Entity.GetLocalBounds

Brief

Retrieve the local-space bounds of an entity's model.

Definition

```
Entity.GetLocalBounds(Entity entity, Vector4 min, Vector4 max)
```

Arguments

entity - The entity to query.min (out) - Vector4 to receive the minimum extents.max (out) - Vector4 to receive the maximum extents.

Return Values

None.

Description

This function will retrieve the local-space bounds of the entity's model. If no model is set then an error will be raised. The returned vectors indicate the minimum and maximum extents on each axis in local space, and define an axis-aligned bounding box. The W components will be 1.

Examples

```
Vector4 min = Vector4.Create()
Vector4 max = Vector4.Create()
myEntity:GetLocalBounds(min, max)
local center = (min + max) * 0.5
local extents = max - center
```

See Also

- Entity.GetWorldBounds

Entity.GetLocalMatrix

Brief

Returns the current local transformation matrix of the entity.

Definition

```
Matrix44 Entity.GetLocalMatrix( Entity entity )
Entity.GetLocalMatrix( Entity entity, Matrix44 outMat )
```

Arguments

entity - The entity to query.outMat - The Matrix44 to store the result to.

Return Values

The current local matrix of the entity.

Description

Returns the current local transformation matrix of the entity. This matrix is relative to the coordinate system of the entity's parent, if one exists. If the entity has no parent, the matrix is relative to the world space coordinate system. If outMat is specified it stores the result in there instead of returning a new Matrix44.

Examples

```
entity = Entity.Create()
mat = Entity.GetLocalMatrix( entity )
print( mat )
```

See Also

- Entity.Create

Entity.GetMass

Brief

Returns the mass of the collision resource attached to the entity.

Definition

```
number Entity.GetMass( Entity entity )
```

Arguments

entity - The entity to query.

Return Values

The mass of the collision resource attached to the entity.

Description

Gets the mass of the rigid body that is attached to the entity. This function can only be called if the entity has a collision resource.

Examples

```
local mass = Entity.GetMass( entity )
```

See Also

- Entity.EnableCollision
- Entity.EnableDynamics
- Entity.SetCollision
- Entity.SetMass

Entity.GetNumActiveAnims

Brief

Returns the number of active animations currently being played on the specified entity.

Definition

```
number Entity.GetNumActiveAnims( Entity entity )
```

Arguments

entity - the entity to query

Return Values

The number of active anims running.

Description

Returns the number of active animations currently being played on the specified entity.

Examples

```
local numAnims = Entity.GetNumActiveAnims( myEntity )
```

See Also

None

Entity.GetParent

Brief

Get the parent entity for the specified entity, if any.

Definition

```
Entity Entity.GetParent(Entity entity)
```

Arguments

entity - entity to obtain parent from.

Return Values

The parent entity, or nil if no parent.

Description

If an entity has been attached to another entity using Entity.AttachToParent, the parent entity can be retrieved with this function.

Examples

```
parent = Entity.Create()
entity = Entity.Create()
entity:AttachToParent(parent)
if (entity:GetParent() == parent) then
    print("Attached!")                    -- prints Attached!
end
```

See Also

- Entity.AttachToParent

- Entity.GetChildren
- Entity.GetSiblings

Entity.GetPointVelocity

Brief

Returns the velocity of a point on a physics collision item.

Definition

```
Vector4 Entity.GetPointVelocity( Entity entity, Vector4 point )
Entity.GetPointVelocity( Entity entity, Vector4 point, Vector4 outVec )
```

Arguments

entity - The entity to query.point - The point of which you wish to know the velocity.outVec - The Vector4 to store the result to.

Return Values

The velocity of the point on the rigid body.

Description

Returns the velocity of a point on a physics collision item in world space. If outVec is specified it stores the result it is used to store the result instead of returning a new Vector4. This function is only available if the entity has a collision item.

Examples

```
print( "Point velocity 1m up = " .. Entity.GetPointVelocity( entity, Vector4.Create( 0, 1, 0 ) )
```

See Also

- Entity.EnableDynamics

Entity.GetPosition

Brief

Returns the entity's position.

Definition

```
Vector4 Entity.GetPosition( Entity entity )
Entity.GetPosition( Entity entity, Vector4 outVec )
```

Arguments

entity - The entity to query.outVec - The Vector4 to store the result to.

Return Values

The entity's position.

Description

Returns the entity's position. If the entity has a parent, the returned position is relative to the parent's coordinate system. If outVec is specified it stores the result in there instead of returning a new Vector4.

Examples

```
print( "Entity position = " .. Entity.GetPosition( entity ) )
```

See Also

None

Entity.GetRotationQuat

Brief

Gets the rotation as a quaternion.

Definition

```
Quaternion Entity.GetRotationQuat( Entity entity )
Entity.GetRotationQuat( Entity entity, Quaternion outQuat )
```

Arguments

entity - The entity to query.outQuat - The Quaternion to store the result to.

Return Values

The rotation of the entity in quaternion form, or none using the second syntax.

Description

Gets the rotation of the entity in quaternion form. If outQuat is specified it is used for storing the result instead of returning a new Quaternion.

Examples

```
entity = Entity.Create()
rotation = entity:GetRotationQuat()
```

See Also

- Entity.SetRotationQuat

Entity.GetRotationXyz

Brief

Gets the rotation about the x, y, and z-axis in degrees.

Definition

```
Vector4 Entity.GetRotationXyz( Entity entity )
Entity.GetRotationXyz( Entity entity, Vector4 outVec )
```

Arguments

entity - The entity to query.outVec - The Vector4 to store the result to.

Return Values

The XYZ-ordered axis rotation of the entity.

Description

Gets the rotation about the x, y, and z-axis of the entity's local coordinate system in degrees. The rotation order is x, y, z. The W component of the returned vector is set to 0.0. If outVec is specified it stores the result in there instead of returning a new Vector4.

Examples

```
entity = Entity.Create()
rotation = Entity.GetRotationXyz( entity )
```

See Also

- Entity.SetRotationXyz

Entity.GetScale

Brief

Returns the entity's scale.

Definition

```
Vector4 Entity.GetScale( Entity entity )
Entity.GetScale( Entity entity, Vector4 outVec )
```

Arguments

entity - The entity to query.outVec - The Vector4 to store the result to.

Return Values

The entity's scale.

Description

Returns the entity's scale. If the entity has a parent, the returned scale is relative to the parent's coordinate system. If outVec is specified it stores the result in there instead of returning a new Vector4.

Examples

```
print( "Entity scale = " .. Entity.GetScale( entity ) )
```

See Also

- Entity.AttachToParent
- Entity.SetScale

Entity.GetSiblings

Brief

Get an array of the sibling entities of a specified entity.

Definition

```
table Entity.GetSiblings(Entity entity)
```

Arguments

entity - entity to obtain siblings from.

Return Values

An array of the sibling entities of the specified entity.

Description

If an entity has been attached to a parent entity using Entity.AttachToParent, this function will return an array of all other entities attached to the same parent, excluding this entity. If this entity is the only child, the table will be empty.

Examples

```
parent = Entity.Create()
child1 = Entity.Create()
child2 = Entity.Create()
child1:AttachToParent(parent)
child2:AttachToParent(parent)
child1Siblings = child1:GetSiblings()
child2Siblings = child2:GetSiblings()
if (child1Sublings[1] ~= child2 or child2Siblings[1] ~= child1) then
    error("This will never happen")
end
```

See Also

- Entity.AttachToParent
- Entity.GetParent
- Entity.GetChildren

Entity.GetUniqueId

Brief

Returns a unique integer ID that identifies the entity.

Definition

```
number Entity.GetUniqueId(Entity entity)
```

Arguments

entity - The entity to query.

Return Values

A unique integer ID value.

Description

The integer ID returned is guaranteed to be unique for all currently active entities on the local client, and it is guaranteed that if entity1 == entity2 is true, then entity1:GetUniqueId() == entity2:GetUniqueId() is also true, and vice versa. No assumptions should be made about the nature of the ID value, and it is not guaranteed to be persistent or reproducible across instances or PS Home versions. Note that the ID value isn't replicated in

any way, it only applies to local entities.

Examples

```
local hitEntity = raycast:GetHitEntity()
if (myEntityTable[ hitEntity:GetUniqueId() ] == nil) then
    print("New entity hit!")
    myEntityTable[ hitEntity:GetUniqueId() ] = hitEntity
end
```

See Also

None

Entity.GetWorldBounds

Brief

Retrieve the world-space bounds of an entity's model.

Definition

```
Entity.GetWorldBounds(Entity entity, Vector4 min, Vector4 max)
```

Arguments

entity - The entity to query.min (out) - Vector4 to receive the minimum extents.max (out) - Vector4 to receive the maximum extents.

Return Values

None.

Description

This function will retrieve the world-space bounds of the entity's model. If no model is set then an error will be raised. The returned vectors indicate the minimum and maximum extents on each axis in world space, and define a world-space axis-aligned bounding box. The W components will be 1. NOTE: the world bounds are only updated when the model renders. Thus if you move the object in current frame, the world bounds will be out of date until the following frame. If you wish to compute up to date world bounds in this specific case, transform the local bounds and take the AABB of the OBB thus formed.

Examples

```
Vector4 min = Vector4.Create()
Vector4 max = Vector4.Create()
myEntity:GetWorldBounds(min, max)
local center = (min + max) * 0.5
local extents = max - center
```

See Also

- Entity.GetLocalBounds

Entity.GetWorldMatrix

Brief

Returns the current world matrix of the entity.

Definition

```
Matrix44 Entity.GetWorldMatrix( Entity entity )
Entity.GetWorldMatrix( Entity entity, Matrix44 outMat )
```

Arguments

entity - The entity to query.outMat - The Matrix44 to store the result to.

Return Values

The current world matrix of the entity.

Description

Returns the current world matrix of the entity. If outMat is specified it stores the result in there instead of returning a new Matrix44.

Examples

```
entity = Entity.Create()
mat = Entity.GetWorldMatrix( entity )
print( mat )
```

See Also

- Entity.Create

Entity.HasActiveAnims

Brief

Indicates if there are any active animations on the entity.

Definition

```
boolean Entity.HasActiveAnims( Entity entity )
```

Arguments

entity - the entity to query

Return Values

True if there are any active animations being played on the entity, or false if not.

Description

This function is more efficient than checking if Entity.GetNumActiveAnims() is 0.

Examples

```
if ( Entity.HasActiveAnims( myEntity ) ) then
    Entity.StopAnim( myEntity )
end
```

See Also

- Entity.BlendAnimIn

- Entity.BlendAnimOut
- Entity.GetIndexOfAnim
- Entity.GetActiveAnim

Entity.IsActionEnabled

Brief

Query whether entity action is enabled.

Definition

```
boolean Entity.IsActionEnabled( Entity entity, number actionId )
```

Arguments

entity - The entity to which the action belongsactionId - The numerical action ID

Return Values

true if enabled, false otherwise

Description

Query whether an entity action is currently enabled.

Examples

```
entity = Entity.Create()
e:SetCollision("kCollisionResource")
e:EnableDynamics()
actionId = Entity.CreatePointForceAction( e, false )
f = Vector4.Create( 10.0, 0.0, 0.0 )
p = Vector4.Create( 0.0, 1.0, 0.0 )
Entity.EnableAction( e, actionId, true )
print( Entity.IsActionEnabled( e, actionId ) )
Entity.SetActionForce( e, actionId, f )
Entity.SetActionPosition( e, actionId, p )
...
Entity.DestroyAction( e, actionId )
```

See Also

- Entity.EnableAction

Entity.IsDynamicsActive

Brief

Query whether dynamics is active on the specified entity.

Definition

```
boolean Entity.IsDynamicsActive( Entity entity )
```

Arguments

entity - The entity to update.

Return Values

true if dynamics is active, false otherwise.

Description

Query whether dynamics is active on the specified entity. See Entity.SetDynamicsActive documentation for more information. This function is only valid if the entity has a collision item.

Examples

```
-- Activate dynamics (collision must be enabled as well)
Entity.EnableCollision( entity )
Entity.EnableDynamics( entity )
Entity.SetDynamicsActive( entity )
print( Entity.IsDynamicsActive( entity ) )
```

See Also

- Entity.EnableDynamics
- Entity.SetDynamicsActive

Entity.IsDynamicsAutoDeactivationEnabled

Brief

Query whether auto-deactivation of dynamics is enabled on the specified entity.

Definition

```
boolean Entity.IsDynamicsAutoDeactivationEnabled( Entity entity )
```

Arguments

entity - The entity to update.

Return Values

true if enabled, false otherwise.

Description

Query whether auto-deactivation of dynamics is enabled on the specified entity. See Entity.EnableDynamicsAutoDeactivation for more information.

Examples

```
-- Enable auto-deactivation
Entity.EnableCollision( entity )
Entity.EnableDynamics( entity )
Entity.SetDynamicsActive( entity )
Entity.EnableDynamicsAutoDeactivation( entity )
print( ( Entity.IsDynamicsAutoDeactivationEnabled( entity ) ) )
```

See Also

- Entity.EnableDynamics

- Entity.SetDynamicsActive
- Entity.EnableDynamicsAutoDeactivation

Entity.IsInterpolatedModeEnabled

Brief

Query whether or not interpolated mode is enabled.

Definition

```
boolean Entity.IsInterpolatedModeEnabled( Entity entity )
```

Arguments

entity - The entity.

Return Values

true if interpolated mode is enabled, false otherwise.

Description

Query whether or not interpolated mode is enabled. Interpolated mode can help smooth the motion of dynamic entities, particularly those that are directly controlled by the user; however, there is an extra processing overhead.

Examples

```
print( Entity.IsInterpolatedModeEnabled( entity ) )
```

See Also

- Entity.EnableInterpolatedMode

Entity.IsKeepUprightEnabled

Brief

Query whether or not the entity is being kept upright.

Definition

```
boolean Entity.IsKeepUprightEnabled( Entity entity )
```

Arguments

entity - entity to be queried.

Return Values

true if the entity is being kept upright, false otherwise.

Description

Query whether or not the entity is being kept upright. e = Entity.Create() e:SetCollision("kCollisionResource") e:EnableCollision() e:EnableDynamics() e:EnableKeepUpright(true, 0.1) print(e:IsKeepUprightEnabled())

Examples

```
Entity.EnableKeepUpright
```

See Also

None

Entity.IsLayerCollisionEnabled

Brief

Query whether or not an entity can collide with collision of the specified layer.

Definition

```
boolean Entity.IsLayerCollisionEnabled( Entity entity, enum layer )
```

Arguments

entity - The entity.layer - The collision layer (a CollisionLayer enum).

Return Values

true if collision is enabled, false otherwise.

Description

Query whether or not an entity can collide with collision of the specified layer.

Examples

```
print( Entity.IsLayerCollisionEnabled( entity1, Entity.GetCollisionLayer( entity2 ) ) )
```

See Also

- Entity.EnableLayerCollision

Entity.IsVisible

Brief

Indicates if the entity is currently marked as visible.

Definition

```
Entity.IsVisible(Entity entity)
```

Arguments

entity - the entity to query.

Return Values

true if the object is marked as visible, false otherwise.

Description

An entity can be marked as invisible for several reasons: the script can do so using SetVisible, entities attached to a person that is invisible or too close to the camera, and entities that represent scene objects hidden by the client (for example, active furniture during removal). This function will indicate whether an entity has been marked invisible for any reason. It is not possible to tell from the return value why the entity is invisible, but it may be helpful to a script if it is desired to hide other entities/effects when one entity is invisible. It is not required to check this for all entities, as only entities attached to scene objects as indicated above can be affected outside the script's control. Visibility is also inherited, so children of an invisible entity are automatically made invisible also, and this functionality should be leveraged if it is desired to hide groups of entities when a parent entity is made invisible for any reason.

Examples

```
if (not Active.GetEntity():IsVisible()) then
    -- active has been hidden, pause our special unattached particle effects
    PauseParticles()
end
```

See Also

- Active.GetEntity
- Entity.SetVisible

Entity.LookAt

Brief

Rotate an entity to look at a specified point.

Definition

```
Entity.LookAt( Entity entity, Vector4 targetPoint, Vector up=[0,1,0,0] )
```

Arguments

entity - The entity to update.targetPoint - The point to look at, in the entity's parent space.up - The up vector of the entity, in the entity's parent space.

Return Values

None.

Description

Rotate the entity to look at the specified point. The entity looks down its forward/front vector, the z-axis. Note that this is a one-off reorientation of the entity, it will not track the look-at point if the entity (or its parent) moves.

Examples

```
entity = Entity.Create()
Entity.LookAt( entity, Vector4.Create( 0.5, 0.5, 0 ) )
```

See Also

- Entity.Create
- Vector4.Create
- Entity.LookAtWorld

Entity.LookAtWorld

Brief

Rotate an entity to look at a specified point in world space.

Definition

```
Entity.LookAtWorld( Entity entity, Vector4 targetPoint, Vector up=[0,1,0,0] )
```

Arguments

entity - The entity to update.targetPoint - The point to look at in world space.up - The up vector of the entity in world space.

Return Values

None.

Description

Rotate the entity to look at the specified point in world space. The entity looks down its forward/front vector, the z-axis. Note that this is a one-off reorientation of the entity, it will not track the look-at point if the entity (or its parent) moves.

Examples

```
entity = Entity.Create()
Entity.LookAtWorld( entity, Vector4.Create( 0.5, 0.5, 0 ) )
```

See Also

- Entity.Create
- Vector4.Create
- Entity.LookAt

Entity.ParticleIsAlive

Brief

Returns true if particle effect is still alive.

Definition

```
boolean Entity.ParticleIsAlive( Entity entity )
```

Arguments

entity - The entity to query.

Return Values

true if any particles are still active, false otherwise.

Description

Returns true if particle effect is still alive.

Examples

```
entity = Entity.Create()
Entity.ParticleSetEffect( entity, "myparticle" )
if Entity.ParticleIsAlive( entity ) then
    print( "Particle system still active!" )
end
```

See Also

- Entity.Create
- Entity.ParticleSetEffect

Entity.ParticleReset

Brief

Resets the particle effect.

Definition

```
Entity.ParticleReset( Entity entity )
```

Arguments

entity - The entity to update.

Return Values

None.

Description

Resets the particle effect by clearing the particle buffer and resetting time to 0.0.

Examples

```
entity = Entity.Create()
Entity.ParticleSetEffect( entity, "myparticle" )
Entity.ParticleReset( entity )
```

See Also

- Entity.Create
- Entity.ParticleSetEffect

Entity.ParticleSetAttachType

Brief

Sets how the particle is attached to the entity's world matrix.

Definition

```
Entity.ParticleSetAttachType( Entity entity, string type )
```

Arguments

entity - The entity to update.type - The type of attachment the entity has over the particle effect. Can be one of the following:
"render"
"emitter"

Return Values

None.

Description

Sets how the particle is attached to the entity's world matrix. Can be one of the following: "render" - The entity's world matrix is used as the render matrix of the particle effect. This is the default. "emitter" - The entity's world matrix is used as the emitter matrix of the particle effect.

Examples

```
e = Entity.Create()
Entity.ParticleSetEffect( e, "flaming_torch" )
Entity.ParticleSetAttachType( e, "emitter" )
```

See Also

- Entity.ParticleSetEffect

Entity.ParticleSetEffect

Brief

Sets the entity's particle effect.

Definition

```
Entity.ParticleSetEffect( Entity entity, string resourceName )
Entity.ParticleSetEffect( Entity entity, Resource particleResource )
```

Arguments

entity - The entity to update.resourceName - The resource name of the particle effect or nil to remove it.particleResource - The particle effect resource or nil to remove it.

Return Values

None.

Description

Sets the entity's particle effect to the named resource as specified in the HDK's Object Editor application.

Examples

```
entity = Entity.Create()
Entity.ParticleSetEffect( entity, "myparticle" )
```

See Also

None

Entity.ParticleSetEmitting

Brief

Starts/Stops particle emission.

Definition

```
Entity.ParticleSetEmitting( Entity entity, boolean value )
```

Arguments

entity - The entity to update.value - true to enable emission and false to stop emission.

Return Values

None.

Description

Starts/Stops particle emission.

Examples

```
entity = Entity.Create()
Entity.ParticleSetEffect( entity, "myparticle" )
Entity.ParticleSetEmitting( entity, false )
```

See Also

- Entity.Create

Entity.ParticleSetUpdate

Brief

Starts/stops particle updating.

Definition

```
Entity.ParticleSetUpdate( Entity entity, boolean value )
```

Arguments

entity - The entity to update.value - true to enable particle updates and false to disable them.

Return Values

None.

Description

Starts/stops particle updating. This effectively starts/stops time.

Examples

```
entity = Entity.Create()
Entity.ParticleSetEffect( entity, "myparticle" )
Entity.ParticleSetUpdate( entity, false )
```

See Also

- Entity.Create
- Entity.ParticleSetEffect

Entity.ParticleSetUserParam

Brief

Sets a user parameter in the particle system

Definition

```
boolean Entity.ParticleSetUserParam( Entity entity, string paramName, number paramValue )
boolean Entity.ParticleSetUserParam( Entity entity, string paramName, Vector4 paramValue )
```

Arguments

entity - The entity to update.paramName - The name of the parameter to set in the particle graph.paramValue - The value to set.

Return Values

true if the assignment was made. false if parameter not present in particle system.

Description

The particle editor tool allows for input of user parameters from runtime code. Access this via the "HomeLua..." nodes in the particle editor. There are many supported types. 1->4 float components. (Scalar, Scale, Vector, Colour, etc...) The Scalar types can use the number or Vector4 interface. All other types need to use the Vector4 interface.

Examples

```
entity = Entity.Create()
Entity.ParticleSetEffect( entity, "myparticle" )
Entity.ParticleSetUserParam( entity, "homelua_scalar1", 5.0 )
Entity.ParticleSetUserParam( entity, "homelua_scalar2", Vector4(5.0) )
Entity.ParticleSetUserParam( entity, "homelua_colour1", Vector4(0.3,0.4,0.5,1.0) )
Entity.ParticleSetUserParam( entity, "homelua_vector5", Vector4(1.0,0.0,1.0) )
```

See Also

None

Entity.PauseAnim

Brief

Pauses all current animation(s) if any are playing.

Definition

```
Entity.PauseAnim( Entity entity )
```

Arguments

entity - The entity to update.

Return Values

None.

Description

Pauses all current animation(s) if any are playing. Use ResumeAnim to continue to play the animation(s).

Examples

```
entity = Entity.Create()
Entity.SetModel( entity, "mymodel" )
Entity.SetSkeleton( entity, "myskel" )
Entity.PlayAnim( entity, "myanim" )
Entity.PauseAnim( entity )
```

See Also

- Entity.ResumeAnim

Entity.PlayAnim

Brief

Plays the specified animation resource.

Definition

```
Entity.PlayAnim( Entity entity, string resourceName, boolean looping = true, number time = 0, number
priority = 0 )
Entity.PlayAnim( Entity entity, Resource animResource, boolean looping = true, number time = 0, number
priority = 0 )
```

Arguments

entity - The entity to update.resourceName - The resource name of the animation.animResource - The animation resource.looping - flag indicating if animation should loop, or only play once. Default is true.time - the time at which the animation is played from. If the time is greater than the duration, is taken as 0. Default is 0.priority - The priority affects where in the animation list an animation will be played. The higher the number, the later it will be played.
If a prior animation is being played at the same priority, then the new animation will be placed after the old animation in the list. (default is 0).
*NB Internally, priority is treated as an integer.

Return Values

None.

Description

Plays the specified animation resource as specified in the HDK's Object Editor application. Stops all other animations that are currently playing.

Examples

```
entity = Entity.Create()
Entity.SetModel( entity, "mymodel" )
Entity.SetSkeleton( entity, "myskel" )
Entity.PlayAnim( entity, "myanim" )
```

See Also

- Entity.Create

- Entity.SetModel
- Entity.SetSkeleton

Entity.RegisterCollisionCallback

Brief

Register a function to be called in response to a collision event involving the specified entity.

Definition

```
Entity.RegisterCollisionCallback( Entity entity, EntityCollision eventType, function callback,
userData = nil )
```

Arguments

entity - the entity to register for the collision callbackeventType - an EntityCollision enum value:
EntityCollision.ContactAdded - a new contact point has been added
EntityCollision.ContactProcess - called for all active contact points each frame while they exist
EntityCollision.ContactRemoved - a contact point has been removedcallback - the Lua function to be calleduserData - arbitrary user data of the user's choice

Return Values

None.

Description

Register a function to be called in response to a collision event involving the specified entity. You must enable collision callbacks for the specified entity using EnableCollisionCallbacks() in order to begin receiving callbacks to your registered function(s). You must also set the maximum contacts buffer using Entity.SetMaxContacts(). Finally, the entity must have collision set and enabled for the callbacks to occur.

Examples

```
-- set up a collision data buffer
Entity.SetMaxContacts( 128 )

foo = Entity.Create()
foo:SetModel("model")
foo:SetCollision("collision")
foo:SetPosition(Vector4.Create(0, 5, 0))
foo:SetVisible(true)
foo:EnableCollision(true)
foo:EnableDynamics(true)
foo:EnableCollisionCallbacks(true)
foo:RegisterCollisionCallback(EntityCollision.ContactAdded, ContactAddedCallback, foo)
foo:RegisterCollisionCallback(EntityCollision.ContactRemoved, ContactRemovedCallback, foo)
```

See Also

- Entity.RegisterCollisionCallback
- Entity.SetMaxContacts
- Entity.SetCollision
- Entity.EnableCollision

Entity.Reset

Brief

Reset an entity to a newly-created state.

Definition

```
Entity.Reset(Entity entity)
```

Arguments

entity - Entity to reset.

Return Values

None.

Description

Resets all non-hierarchy entity state, including model, animation, collision, particles, attached cameras, lights and other state. The entity's hierarchy state (parent and children) and transform state (position and rotation) will not be reset, if this is desired it must be done explicitly. Of particular use if using a pool of entities, as entities can be reset upon being released back into the pool (or when allocated from the pool).

Examples

```
myEntity:Reset()
```

See Also

- Entity.Copy

Entity.ResumeAnim

Brief

Resumes all current animation(s) if they are paused.

Definition

```
Entity.ResumeAnim( Entity entity )
```

Arguments

entity - The entity to update.

Return Values

None.

Description

Resumes all current animation(s) if they are paused. Use PauseAnim to pause the animation(s).

Examples

```
entity = Entity.Create()
Entity.SetModel( entity, "mymodel" )
Entity.SetSkeleton( entity, "myskel" )
Entity.PlayAnim( entity, "myanim" )
Entity.PauseAnim( entity )
Entity.ResumeAnim( entity )
```

See Also

- Entity.PauseAnim

Entity.SetActionForce

Brief

Set the force of an entity action.

Definition

```
Entity.SetActionForce( Entity entity, number actionId, Vector4 force )
```

Arguments

entity - The entity to which the action belongsactionId - The numerical action IDforce - The force in world-space

Return Values

None.

Description

Set the force of an entity action. Only applicable to force-based actions. This function is only available if the entity has a collision item.

Examples

```
entity = Entity.Create()
e:SetCollision("kCollisionResource")
e:EnableDynamics()
actionId = Entity.CreatePointForceAction( e, false )
f = Vector4.Create( 10.0, 0.0, 0.0 )
p = Vector4.Create( 0.0, 1.0, 0.0 )
Entity.EnableAction( e, actionId )
Entity.SetActionForce( e, actionId, f )
Entity.SetActionPosition( e, actionId, p )
...
Entity.DestroyAction( e, actionId )
```

See Also

- Entity.SetActionTorque
- Entity.SetActionPosition

Entity.SetActionPosition

Brief

Set the position of an entity action.

Definition

```
Entity.SetActionPosition( Entity entity, number actionId, Vector4 position )
```

Arguments

entity - The entity to which the action belongsactionId - The numerical action IDposition - The position in world-space.

Return Values

None.

Description

Set the position of an entity action. Only applicable to position-based actions. This function is only available if the entity has a collision item.

Examples

```
entity = Entity.Create()
e:SetCollision("kCollisionResource")
e:EnableDynamics()
actionId = Entity.CreatePointForceAction( e, false )
f = Vector4.Create( 10.0, 0.0, 0.0 )
p = Vector4.Create( 0.0, 1.0, 0.0 )
Entity.EnableAction( e, actionId )
Entity.SetActionForce( e, actionId, f )
Entity.SetActionPosition( e, actionId, p )
...
Entity.DestroyAction( e, actionId )
```

See Also

- Entity.SetActionTorque
- Entity.SetActionForce

Entity.SetActionTorque

Brief

Set the torque of an entity action.

Definition

```
Entity.SetActionTorque( Entity entity, number actionId, Vector4 torque )
```

Arguments

entity - The entity to which the action belongsactionId - The numerical action IDtorque - The torque in world-space.

Return Values

None.

Description

Set the torque of an entity action. Only applicable to torque-based actions. This function is only available if the entity has a collision item.

Examples

```
entity = Entity.Create()
e:SetCollision("kCollisionResource")
e:EnableDynamics()
actionId = Entity.CreateTorqueAction( e, false )
t = Vector4.Create( 10.0, 0.0, 0.0 )
p = Vector4.Create( 0.0, 1.0, 0.0 )
Entity.EnableAction( e, actionId )
Entity.SetActionTorque( e, actionId, t )
...
Entity.DestroyAction( e, actionId )
```

See Also

- Entity.SetActionForce
- Entity.SetActionPosition

Entity.SetAngularDamping

Brief

Sets the angular damping of the collision resource attached to the entity.

Definition

```
Entity.SetAngularDamping( Entity entity, number damping )
```

Arguments

entity - The entity to update.damping - The angular damping to apply to the entity.

Return Values

None.

Description

Sets the angular damping of the collision resource attached to the entity. Angular damping reduces the rotational velocity of the entity over time. For example, if you set the angular damping to 0.2, then every second about 20% of the object's angular velocity will be removed. This function is only valid if the entity has a collision item.

Examples

```
Entity.SetAngularDamping( entity, 0.2 )
```

See Also

- Entity.EnableCollision
- Entity.EnableDynamics
- Entity.SetCollision
- Entity.GetAngularDamping

Entity.SetAngularVelocity

Brief

Sets the angular velocity around the center of mass of the entity.

Definition

```
Entity.SetAngularVelocity( Entity entity, Vector4 velocity )
```

Arguments

entity - The entity to update.velocity - vector specifying angular velocity.

Return Values

None.

Description

Sets the angular velocity around the center of mass of the entity's rigid body in world space. The W component of the specified vector is ignored.

Examples

```
-- Add the entity to the physics simulation
Entity.EnableCollision( entity, true )
Entity.EnableDynamics( entity, true )

Entity.SetAngularVelocity( entity, Vector4.Create( 0, 10, 0 ) )
```

See Also

- Entity.SetLinearVelocity
- Vector4.Create

Entity.SetAttachType

Brief

Set the type of attachment used by the entity.

Definition

```
Entity.SetAttachType( Entity entity, AttachType type )
```

Arguments

entity - The entity to affect (must be attached to another entity or person).type - The type of attachment to use:
AttachType.Translation
AttachType.Rotation
AttachType.Scale
AttachType.Transform (default)

Return Values

None.

Description

The default behaviour for attached entities is to inherit all of their parent's transform, including translation, rotation and scaling. A script can use this function to only inherit some of the parent's transform, to only translate an object with its parent and not alter its rotation for example. Scale is handled so that if translation and scale are both inherited, then the local translation of the child entity will be scaled also, as per normal attachment. If translation is not inherited, the local translation will not be scaled.

Examples

```
child:AttachToParent(parent)
child:SetAttachType(AttachType.Translation)                         -- only inherit parent translation
child:SetAttachType(AttachType.Translation + AttachType.Rotation)   -- ignore parent scale
child:SetAttachType(AttachType.Transform)                           -- same as default
```

See Also

- Entity.AttachToParent
- Entity.AttachToBone

Entity.SetCenterOfMassLocal

Brief

Sets the position of the entity's center of mass in local space.

Definition

```
Entity.SetCenterOfMassLocal( Entity entity, Vector4 com )
```

Arguments

entity - The entity to update.com - The position of the entity's center of mass in local space.

Return Values

None.

Description

Sets the position of the entity's center of mass in local space. This may be offset from the coordinate origin of the entity depending on its physics configuration. This function is only available if the entity has a collision item.

Examples

```
Entity.SetCenterOfMassLocal( entity, Vector4.Create( 0.5, 0.5, 0.5 ) )
```

See Also

- Entity.EnableDynamics

Entity.SetCollision

Brief

Sets the entity's collision.

Definition

```
Entity.SetCollision( Entity entity, string resourceName, Phys3dWorld world = nil )
Entity.SetCollision( Entity entity, Resource collisionResource, Phys3dWorld world = nil )
```

Arguments

entity - The entity to update.resourceName - The collision's resource name or nil to remove it.collisionResource - The collision resource or nil to remove it.world - Phys3dWorld to add the entity to, or nil for the default.

Return Values

None.

Description

Sets the entity's collision to the named resource as specified in the HDK's Object Editor application. Note that an entity with collision will no longer respect attachments specified prior to calling SetCollision. The current world transform of the entity is preserved across this function call. This function will automatically enable collision on the entity.

Examples

```
entity = Entity.Create()
Entity.SetCollision( entity, "mycollision" )

-- Remove the collision from the entity
Entity.SetCollision( entity, nil )
```

See Also

- Entity.Create
- Phys3dWorld.Create

Entity.SetCollisionLayer

Brief

Set the collision layer of an entity.

Definition

```
Entity.SetCollisionLayer( Entity entity, enum layer, boolean preserveCurrentLayerCollisions = true )
```

Arguments

entity - The entity.layer - The collision layer; can be either CollisionLayer.Entity or a "User Layer".preserveCurrentLayerCollisions - true to preserve which layers the entity collides with, false otherwise.

Return Values

None.

Description

Set the collision layer of an entity. If preserveCurrentLayerCollisions is false, the layers with which the entity collides are reset to the default for the specified layer. The defaults can be changed using Scene.SetLayerCollisionDefault.

Examples

```
Entity.SetCollisionLayer( entity, CollisionLayer.User1, false )
```

See Also

- Entity.GetCollisionLayer
- Scene.SetLayerCollisionDefault

Entity.SetCullDistance

Brief

Set a distance at which the entity will be automatically made invisible.

Definition

```
Entity.SetCullDistance(Entity entity, number distance)
```

Arguments

entity - The entity to modify.distance - The distance from camera at which to visibility cull the entity, or 0 to disable.

Return Values

None.

Description

For situations involving a large scene, it may be beneficial for efficiency reasons to set objects to invisible when they are located far away. Setting a cull distance will automatically compute the entity's distance from camera every frame and hide it and any child entities if the distance is greater than the specified cull distance. Note that internally squared distances are compared, so the cull distance should not overflow a floating point number when squared.

Examples

```
myEntity:SetCullDistance(80)        -- set cull distance to 80m
```

See Also

- Entity.GetCullDistance

Entity.SetDynamicsActive

Brief

Activate/deactivate dynamics on the specified entity.

Definition

```
Entity.SetDynamicsActive( Entity entity, boolean activate = true )
```

Arguments

entity - The entity to update.activate - true to activate dynamics, false otherwise (defaults to true if not specified).

Return Values

None.

Description

Dynamics must be enabled with Entity.EnableDynamics() for the activated/deactivated state to take effect. An Entity with activated dynamics is one that is having it's motion updated. If dynamics are deactivated with this function, Home's physics engine will stop updating the Entity's motion, until it is activated again. Entities can be activated by another call to this function or by the application of forces/torques/impulses or by coming into contact with other activated Entities. Entities can be deactivated with this function or by allowing auto-deactivation which will deactivate the Entity if it has not been moving for a while. Auto-deactivation can be enabled/disabled with Entity.EnableDynamicsAutoDeactivation. This function is only valid if the entity has a collision item.

Examples

```
-- Deactivate dynamics
Entity.SetDynamicsActive( entity, false )

-- Activate dynamics (collision must be enabled as well)
Entity.EnableCollision( entity )
Entity.EnableDynamics( entity )
Entity.SetDynamicsActive( entity )
```

See Also

- Entity.EnableDynamics
- Entity.EnableDynamicsAutoDeactivation

Entity.SetFrictionCoeff

Brief

Set the friction coefficient of the collision resource attached to the entity.

Definition

```
Entity.SetFrictionCoeff( Entity entity, number f )
```

Arguments

entity - The entity to update.f - the coefficient of friction.

Return Values

None.

Description

Set the coefficient of friction for the surface of the rigid body that is attached to the entity. This affects the entity's motion relative to surfaces it comes into contact with. Although this is actually a pair-wise coefficient, friction is specified on a per rigid body basis. When affecting the relative motion of two surfaces, the surfaces' friction coefficients are combined to produce a coefficient for the interaction. Usual values for friction are between 0 and 1. Values greater than 1, although physically invalid, are allowed. However, values must be less than or equal to 255.

Examples

```
Entity.SetFrictionCoeff( entity, 0.0 )
```

See Also

- Entity.EnableCollision
- Entity.EnableDynamics
- Entity.SetCollision

Entity.SetGravity

Brief

Set the gravity vector to apply to the entity.

Definition

```
Entity.SetGravity( Entity entity, Vector4 gravityVector )
```

Arguments

entity - entity to be affected.gravityVector - new gravity vector.

Return Values

None.

Description

Only the specified entity is affected by the new gravity force.

Examples

```
-- set gravity to half strength
local g = Scene.GetGravity() * 0.5
entity:SetGravity( g )
```

See Also

- Entity.GetGravity

Entity.SetLight

Brief

Set the entity's light.

Definition

```
Entity.SetLight( Entity entity, Light light = nil )
```

Arguments

entity - The entity to update.light - The Light to attach to the entity or nil to detach the existing light.

Return Values

None.

Description

Set the entity's light.

Examples

```
entity = Entity.Create()
Entity.SetLight( entity, light )
```

See Also

- Entity.Create

Entity.SetLinearDamping

Brief

Sets the linear damping of the collision resource attached to the entity.

Definition

```
Entity.SetLinearDamping( Entity entity, number damping )
```

Arguments

entity - The entity to update.damping - The linear damping to apply to the entity.

Return Values

None.

Description

Sets the linear damping of the collision resource attached to the entity. Linear damping reduces the linear velocity of the entity over time. For example, if you set the linear damping to 0.2, then every second about 20% of the object's linear velocity will be removed. This function is only valid if the entity has a collision item.

Examples

```
Entity.SetLinearDamping( entity, 0.2 )
```

See Also

- Entity.EnableCollision
- Entity.EnableDynamics
- Entity.SetCollision
- Entity.GetLinearDamping

Entity.SetLinearVelocity

Brief

Sets the linear velocity of this object.

Definition

```
Entity.SetLinearVelocity( Entity entity, Vector4 velocity )
```

Arguments

entity - The entity to update.velocity - Vector specifying linear velocity.

Return Values

None.

Description

Sets the linear velocity of the center of mass of the entity's rigid body in world space. The W component of the specified vector is ignored.

Examples

```
-- Add the entity to the physics simulation
Entity.EnableCollision( entity, true )
Entity.EnableDynamics( entity, true )

Entity.SetLinearVelocity( entity, Vector4.Create( 0, 10, 0 ) )
```

See Also

- Entity.SetAngularVelocity
- Vector4.Create

Entity.SetLocalMatrix

Brief

Sets the entity's local transformation matrix.

Definition

```
Entity.SetLocalMatrix( Entity entity, Matrix44 matrix )
```

Arguments

entity - The entity to update.matrix - The matrix to set the entity's transform to.

Return Values

None.

Description

Sets the entity's local transformation matrix. The specified matrix is relative to the coordinate system of the entity's parent, if one exists. If the entity has no parent, the matrix is relative to the world space coordinate system.

Examples

```
entity = Entity.Create()
matrix = Matrix44.Create()

-- Set up a uniform scale matrix
Matrix44.SetScale( matrix, 2, 2, 2, 1, MATRIX_REPLACE )
Entity.SetLocalMatrix( entity, matrix )
print( mat )
```

See Also

- Entity.Create
- Matrix44.Create
- Matrix44.SetScale

Entity.SetMass

Brief

Sets the mass of the collision resource attached to the entity.

Definition

```
Entity.SetMass( Entity entity, number mass )
```

Arguments

entity - The entity to update.mass - The mass of the entity's collision resource in kilograms.

Return Values

None.

Description

Sets the mass of the collision resource that is attached to the specified entity. Very large or very small masses should be avoided, as these can create instability in the simulation. This function can only be called if the entity has a collision resource.

Examples

```
Entity.SetMass( entity, 5 )
```

See Also

- Entity.EnableCollision
- Entity.EnableDynamics
- Entity.SetCollision
- Entity.GetMass

Entity.SetMaxContacts

Brief

Specify the maximum number of contacts to buffer for collision callbacks.

Definition

```
boolean Entity.SetMaxContacts(number maxContacts)
```

Arguments

maxContacts - maximum number of contact events to buffer

Return Values

true if sufficient memory could be allocated or false otherwise.

Description

Allocates an internal buffer to record contact events for provision via collision callbacks. Each contact event generated by PS Home for the types for which the script has registered callbacks will be buffered in this internal buffer, until the contacts are passed to the script via the callbacks. All entities registered for callbacks in the current script environment will share this buffer. If more contacts are generated than the buffer can store, subsequent contacts will not be buffered and will not be available to the script, and a message indicating this will be displayed in the console. Note that each contact currently requires 64 bytes of memory. If this function fails, there was insufficient memory available to the script to allocate the buffer. In this case, collision callbacks will be disabled until a subsequent call to this function (with a lower maximum) succeeds. This function is not available from within a collision callback and will generate an error.

Examples

```
Entity.SetMaxContacts(128)      -- enough memory for 128 contact events (added/processed/removed) per
frame
```

See Also

- Entity.RegisterCollisionCallback
- Entity.EnableCollisionCallbacks
- Entity.SetCollision
- Entity.EnableCollision

Entity.SetModel

Brief

Set the entity's model.

Definition

```
Entity.SetModel( Entity entity, string resourceName )
Entity.SetModel( Entity entity, Resource modelResource )
```

Arguments

entity - The entity to update.resourceName - The model's resource name or nil to remove the model.modelResource - The model resource or nil to remove the model.

Return Values

None.

Description

Set the entity's model to the named resource as specified in the HDK's Object Editor application.

Examples

```
entity = Entity.Create()
Entity.SetModel( entity, "mymodel" )

-- Remove the model from the entity
Entity.SetModel( entity, nil )
```

See Also

- Entity.Create

Entity.SetPosition

Brief

Sets the entity's position.

Definition

```
Entity.SetPosition( Entity entity, Vector4 position )
```

Arguments

entity - The entity to update.position - The new position for the entity.

Return Values

None.

Description

Sets the entity's position. If the entity has a parent, the specified position is relative to local coordinate system of the parent.

Examples

```
entity = Entity.Create()
Entity.SetPosition( entity, Vector4.Create( 10, 10, 0 ) )
```

See Also

- Entity.Create

Entity.SetRestitutionCoeff

Brief

Sets the restitution coefficient of the collision resource attached to the entity.

Definition

```
Entity.SetRestitutionCoeff( Entity entity, number r )
```

Arguments

entity - The entity to update.r - the coefficient of restitution.

Return Values

None.

Description

Sets the restitution (or elasticity) coefficient for the rigid body attached to the entity. This governs the effect that collisions have on the velocities of colliding entities. Like friction, this is a pair-wise coefficient - when two physical entities collide, their restitution values are combined to produce a coefficient for the interaction. Usual values for restitution are between 0 and 1. Values greater than 1, although physically invalid, are allowed. However, values must be less than 1.99.

Examples

```
-- Prevent entity from bouncing
Entity.SetRestitutionCoeff( entity, 0.0 )
```

See Also

- Entity.EnableCollision

- Entity.EnableDynamics
- Entity.SetCollision

Entity.SetRotationQuat

Brief

Sets the rotation using the supplied quaternion.

Definition

```
Entity.SetRotationQuat( Entity entity, Quaternion quat )
```

Arguments

entity - The entity to update.quat - The quaternion.

Return Values

None.

Description

Sets the rotation of the specified entity using the supplied quaternion.

Examples

```
entity = Entity.Create()
quat = Quaternion.Create()
entity:SetRotationQuat(quat)
```

See Also

- Entity.GetRotationQuat

Entity.SetRotationX

Brief

Sets the rotation about the x-axis in degrees.

Definition

```
Entity.SetRotationX( Entity entity, number rotation )
```

Arguments

entity - The entity to update.rotation - The specified angle of rotation about the x-axis in degrees.

Return Values

None.

Description

Sets the rotation about the x-axis of the entity's local coordinate system in degrees. The rotation order is x, y, z.

Examples

```
entity = Entity.Create()
Entity.SetRotationX( entity, 90.0 )
```

See Also

- Entity.Create

Entity.SetRotationXyz

Brief

Sets the rotation about the x, y, and z-axis in degrees.

Definition

```
Entity.SetRotationXyz( Entity entity, Vector4 rotation )
```

Arguments

entity - The entity to update.rotation - The specified x, y, and z angles of rotation in degrees.

Return Values

None.

Description

Sets the rotation about the x, y, and z-axis of the entity's local coordinate system in degrees. The rotation order is x, y, z.

Examples

```
entity = Entity.Create()
Entity.SetRotationXyz( entity, Vector4.Create( 45, 30, 90 ) )
```

See Also

- Entity.Create

Entity.SetRotationY

Brief

Sets the rotation about the y-axis in degrees.

Definition

```
Entity.SetRotationY( Entity entity, number rotation )
```

Arguments

entity - The entity to update.rotation - The specified angle of rotation about the y-axis in degrees.

Return Values

None.

Description

Sets the rotation about the y-axis of the entity's local coordinate system in degrees. The rotation order is x, y, z.

Examples

```
entity = Entity.Create()
Entity.SetRotationY( entity, 90.0 )
```

See Also

- Entity.Create

Entity.SetRotationZ

Brief

Sets the rotation about the z-axis in degrees.

Definition

```
Entity.SetRotationZ( Entity entity, number rotation )
```

Arguments

entity - The entity to update.rotation - The specified angle of rotation about the z-axis in degrees.

Return Values

None.

Description

Sets the rotation about the z-axis of the entity's local coordinate system in degrees. The rotation order is x, y, z.

Examples

```
entity = Entity.Create()
Entity.SetRotationZ( entity, 90.0 )
```

See Also

- Entity.Create

Entity.SetScale

Brief

Sets the entity's scale.

Definition

```
Entity.SetScale( Entity entity, Vector4 scale )
```

Arguments

entity - The entity to update.scale - The new scale of the entity.

Return Values

None.

Description

Sets the entity's scale. The specified scale is applied to the local transformation matrix of the entity. On creation, an entity has a scale of (1, 1, 1) or no scaling, in other words. Note that the W component of the specified vector is ignored. Note that scale should not be applied to entities with collision enabled.

Examples

```
entity = Entity.Create()
Entity.SetScale( entity, Vector4.Create( 45, 30, 90 ) )
```

See Also

- Entity.Create
- Vector4.Create

Entity.SetScreen

Brief

Set the entity's screen.

Definition

```
Entity.SetScreen( Entity entity, Screen screen )
```

Arguments

entity - The entity to update.screen - The screen to attach to the entity or nil to remove it.

Return Values

None.

Description

Set the entity's screen. The screen's corner coordinates will be transformed by the entity's world transformation matrix.

Examples

```
entity = Entity.Create()
Entity.SetScreen( entity, screen )

-- Remove the screen from the entity
Entity.SetScreen( entity, nil )
```

See Also

- Entity.Create

Entity.SetSkeleton

Brief

Set the entity's skeleton.

Definition

```
Entity.SetSkeleton( Entity entity, string resourceName )
Entity.SetSkeleton( Entity entity, Resource skeletonResource )
```

Arguments

entity - The entity to update.resourceName - The skeleton's resource name or nil to remove it.skeletonResource - Pre-loaded skeleton resource object.

Return Values

None.

Description

Set the entity's skeleton to the named resource as specified in the HDK's Object Editor application.

Examples

```
entity = Entity.Create()
Entity.SetSkeleton( entity, "myskel" )

-- Remove the skeleton from the entity
Entity.SetSkeleton( entity, nil )
```

See Also

- Entity.Create

Entity.SetVisible

Brief

Sets the entity to visible or invisible.

Definition

```
Entity.SetVisible( Entity entity, boolean value )
```

Arguments

entity - The entity to update.value - If true the entity is visible and if false it is invisible.

Return Values

None.

Description

Sets the entity to visible or invisible. Entities default to visible when created.

Examples

```
entity = Entity.Create()
Entity.SetVisible( entity, false )
```

See Also

- Entity.Create

Entity.StopAnim

Brief

Stops the current animation if it is playing.

Definition

```
Entity.StopAnim( Entity entity )
Entity.StopAnim( Entity entity, Animation anim )
```

Arguments

entity - The entity to update.anim - The animation to stop.

Return Values

None.

Description

If the Animation anim passed in is nil, or invalid, nothing is done. If an animation is specified, it will be stopped immediately with no blend out. If no animation is specified, all animations on the entity will be stopped immediately, with no blend out.

Examples

```
entity = Entity.Create()
local animHandle = Animation.Create()
Entity.SetModel( entity, "mymodel" )
Entity.SetSkeleton( entity, "myskel" )
Entity.PlayAnim( entity, "my1stAnim" )
Entity.BlendAnim( entity, "my2ndAnim", true, 0, 0, animHandle )
...
Entity.StopAnim( entity, animHandle ) -- stops specific animation
Entity.StopAnim( entity ) -- stops all other animations
```

See Also

- Entity.Create
- Entity.PlayAnim
- Entity.SetModel
- Entity.SetSkeleton

Environment

Environment.DebugProfileBegin

Brief

Begins profiling using the built in object profiler.

Definition

```
Environment.DebugProfileBegin( number profileIndex )
```

Arguments

profileIndex - Which user profile to use. Ranges from 1 - 4.

Return Values

None

Description

Begins profiling using the built in object profiler. Stop profiling the section with Environment.DebugProfileEnd.

Examples

```
function Update()
    Environment.DebugProfileBegin( 1 )
    RunAllMyCode()
    Environment.DebugProfileEnd( 1 )
end
```

See Also

- Environment.DebugProfileEnd

Environment.DebugProfileEnd

Brief

Ends the current profiling section.

Definition

```
Environment.DebugProfileEnd( number profileIndex )
```

Arguments

profileIndex - Which user profile to use. Ranges from 1 - 4.

Return Values

None

Description

Ends the current profiling section. This requires calling Environment.DebugProfileBegin with the matching profile index beforehand.

Examples

```
function Update()
    Environment.DebugProfileBegin( 3 )
    RunAllMyCode()
    Environment.DebugProfileEnd( 3 )
end
```

See Also

- Environment.DebugProfileBegin

Environment.EnableCallbackGc

Brief

Enable/Disable automatic garbage collection from callbacks

Definition

```
Environment.EnableCallbackGc( boolean value )
```

Arguments

value - true if you want to enable the callback GC and false to disable it

Return Values

None

Description

By default any lua environment created in Home will be subject to a garbage collection cycle. Callback Gc occurs after X non frame based callbacks, for example OnMessageReceived(), and does not include update callbacks such as OnDecideIfLocalPlayerCanJoinGame(), OnUpdate() and OnLocalPlayerUpdateGamePlay(). The garbage collection operation can take several milliseconds to complete, the cost increasing in proportion to the amount of data being handled by the script so it can be beneficial to control the garbage collection.
Note: See the HDK Lua Scripting User Guide for full details on Garbage Collection.

Examples

```
Environment.EnableCallbackGc( true )            -- enables garbage collection
Environment.EnableCallbackGc( false )           -- disables garbage collection
```

See Also

- Environment.EnableFrameGc
- Environment.SetCallbackGcInterval

Environment.EnableFrameGc

Brief

Enable/Disable per-frame automatic garbage collection

Definition

```
Environment.EnableFrameGc( boolean value)
```

Arguments

value - true if you want to enable the frame GC and false to disable it

Return Values

Description

By default any lua environment created in Home will be subject to a garbage collection cycle. This occurs after every frame. The garbage collection operation can take several milliseconds to complete, the cost increasing in proportion to the amount of data being handled by the script so it can be beneficial to control the garbage collection.
Note: See the HDK Lua Scripting User Guide for full details on Garbage Collection.

Examples

```
Environment.EnableFrameGc( true )       -- enables garbage collection
Environment.EnableFrameGc( false )      -- disables garbage collection
```

See Also

- Environment.EnableCallbackGc

Environment.SetCallbackGcInterval

Brief

Sets the intervals that the callback garbage collector is initiated

Definition

```
Environment.SetCallbackGcInterval(n)
```

Arguments

n - the number of times a callback hook is called before a garbage collection is initiated.

Return Values

None

Description

This interval applies to any non-update related callback hooks (such as network bag or message related callbacks). It is possible for several such callbacks to be made inside one frame which can cause lua script to run out of memory, for example 50 callbacks of OnMessageReceived could happen in a single frame. These callbacks are counted and if they reach the specified interval PS Home will force a garbage collect. If any frame garbage collection occurs it will reset the count so the system will not unnecessarily run more than one garbage collection cycle per frame.
By default the interval has a value of 10.
Note: See the HDK Lua Scripting User Guide for full details on Garbage Collection.

Examples

```
Environment.EnableCallbackGc( true )        -- enable gc
Environment.SetCallbackGcStep( 0 )          -- full garbage collection
Environment.SetCallbackGcInterval( 2 )      -- after two callbacks the system will do a garbage
collect
```

See Also

- Environment.EnableCallbackGc

Environment.SetCallbackGcStep

Brief

Sets the heuristic value used for doing a step of garbage collection per callback update

Definition

```
Environment.SetCallbackGcStep( number step )
```

Arguments

step - 0 for full garbage collection or > 0 for smaller steps.

Return Values

None

Description

The step controls how much the callback garbage collector collects per call (when enabled) and has hit the number of required invervals. The default is for the garbage collector to do a full collect, but by using any number greater than 0 a user can instruct the garbage collector to do smaller steps of work. Larger values equate to more steps during garbage collection. A step value greater than 0 is equivalent to the lua function collectgarbage("step", step). A step value of 0 is equivalent to the lua function collectgarbage("collect").
Note: See the HDK Lua Scripting User Guide for full details on Garbage Collection.

Examples

```
Environment.EnableCallbackGc( true )        -- enable gc
Environment.SetCallbackGcStep( 2 )          -- do a small amount of gc per update
```

See Also

- Environment.EnableCallbackGc
- Environment.SetCallbackGcInterval

Environment.SetFrameGcStep

Brief

Sets the heuristic value used for doing a step of garbage collection per frame

Definition

```
Environment.SetFrameGcStep( number step )
```

Arguments

step - 0 for full garbage collection or > 0 for smaller steps.

Return Values

None

Description

The step controls how much the frame update garbage collector collects per call (when enabled). The default is for the garbage collector to do a full collect, but by using any number greater than 0 a user can instruct the garbage collector to do smaller steps of work. Larger values equate to

more steps during garbage collection. A step value greater than 0 is equivalent to the lua function collectgarbage("step", step). A step value of 0 is equivalent to the lua function collectgarbage("collect").
Note: See the HDK Lua Scripting User Guide for full details on Garbage Collection.

Examples

```
Environment.EnableFrameGc( true )           -- enable gc
Environment.SetFrameGcStep( 2 )             -- do a small amount of gc per update
```

See Also

- Environment.EnableFrameGc

Font

Font.Create

Brief

Create a font object given a font resource and parameters.

Definition

```
Font Font.Create( Resource fontResource, float fontSize, float fontSlant = 0 )
Font Font.Create( string resourceName, float fontSize, float fontSlant = 0 )
```

Arguments

fontResource - The font file resource.resourceName - The name of a font file resource in the object.fontSize - The size for the font (value must be positive).fontSlant - The slant for the font in range -1 to 1 (optional parameter, default is 0).

Return Values

The new font instance.

Description

A font resource is a file resource that references a TTF font file. This function will create a font object from the resource, which must be already loaded, that can be passed to Renderer.SetFont to specify a font used for text rendering. Note that both the font object and its associated resource must continue to exist while the font is in use. To allow the font and resource to be garbage collected, any renderers referencing the font should be changed to another custom font or back to a system font.

Examples

```
font = Font.Create(myFontResource, 8)
myRenderer:SetFont(font)
myRenderer:DrawText2d(textPos, "Look at my custom font!")
```

See Also

- Renderer.SetFont

Furniture

Furniture.ClearLayout

Brief

Remove all furniture from the current personal space.

Definition

```
Furniture.ClearLayout()
```

Arguments

None

Return Values

Description

Remove all furniture from the current personal space. This function should only be called if the furniture system is not busy. This function should only be called if the current scene is one in which the local player is allowed to use the furniture placement system.

Examples

```
if not Furniture.IsSystemBusy() then
    Furniture.ClearLayout()
end
```

See Also

- Furniture.IsSystemBusy

Furniture.ExportLayout

Brief

Export the furniture layout of the current scene to a memory container.

Definition

```
MemoryContainer Furniture.ExportLayout()
```

Arguments

None

Return Values

Description

Export the furniture layout of the current scene to a memory container. This function should only be called if the furniture system is not busy. This function should only be called if the current scene is one in which the local player is allowed to use the furniture placement system.

Examples

```
if not Furniture.IsSystemBusy() then
     mc = Furniture.ExportLayout()
end
```

See Also

- Furniture.IsSystemBusy
- Furniture.ImportLayout

Furniture.GetCategories

Brief

Get information about furniture categories.

Definition

```
table Furniture.GetCategories()
```

Arguments

None

Return Values

A table containing information about furniture categories.

Description

Get information about furniture categories. The table will contain an entry for each furniture category, indexed from 1 to the number of categories. Each entry will be a table with the following members: 'id' - the furniture system unique id for the category. 'name' - the localised name for the category. 'thumb' - a handle to the thumbnail for the category.

Examples

```
local function GetCategoryData()
    local   data = {}
    local   categories = Furniture.GetCategories()
    local   numCategories = #categories
    for i = 1, numCategories do
        data[ categories[i].id ] = {}
        data[ categories[i].id ].name = categories[i].name
        data[ categories[i].id ].thumb = categories[i].thumb
        data[ categories[i].id ].objects = {}
    end
    local   numItems = Furniture.GetItemCount()
    for i = 1, numItems do
        local   itemData = Furniture.GetItemData( i )
        data[ itemData.categoryId ].objects[ itemData.objectId ] = true
    end
    return data
end
```

See Also

- Furniture.GetItemCount
- Furniture.GetItemData

Furniture.GetItemCount

Brief

Get number of furniture items in the current scene.

Definition

```
number Furniture.GetItemCount()
```

Arguments

None

Return Values

The number of furniture items in the current scene.

Description

Get number of furniture items in the current scene. This function should only be called if the furniture system is not busy.

Examples

```
if Furniture.IsSystemBusy() == false then
    local   numItems = Furniture.GetItemCount()
    for i = 1, numItems do
        local   itemData = Furniture.GetItemData( i )
        print( itemData )
    end
end
```

See Also

- Furniture.GetItemData
- Furniture.IsSystemBusy

Furniture.GetItemData

Brief

Get information about an item of furniture in the current scene.

Definition

```
table Furniture.GetItemData( number index )
```

Arguments

index - the index of the furniture item (1 <= index <= numItems)

Return Values

A table containing information about the furniture item.

Description

Get information about an item of furniture in the current scene. The table will contain the following information: 'id' - the furniture system unique id for the item. 'categoryId' - a FurnitureCategory enum. 'objectId' - the object id for the item. 'pos' - the position of the item as a Vector4. 'rot' - the rotation of the item as a Quaternion. This function should only be called if the furniture system is not busy.

Examples

```
if Furniture.IsSystemBusy() == false then
    local   numItems = Furniture.GetItemCount()
    for i = 1, numItems do
        local   itemData = Furniture.GetItemData( i )
        print( itemData )
    end
end
```

See Also

- Furniture.GetItemCount
- Furniture.IsSystemBusy

Furniture.ImportLayout

Brief

Import a furniture layout from a memory container into the current scene.

Definition

```
Furniture.ImportLayout( MemoryContainer container )
```

Arguments

None

Return Values

Description

Import a furniture layout from a memory container into the current scene. This function should only be called if the furniture system is not busy. This function should only be called if the current scene is one in which the local player is allowed to use the furniture placement system.

Examples

```
if not Furniture.IsSystemBusy() then
    Furniture.ImportLayout( mc )
end
```

See Also

- Furniture.IsSystemBusy
- Furniture.ExportLayout

Furniture.IsSystemBusy

Brief

Query whether or not the furniture system is busy.

Definition

```
boolean Furniture.IsSystemBusy()
```

Arguments

None

Return Values

true if the furniture system is busy, false otherwise.

Description

Query whether or not the furniture system is busy.

Examples

```
if Furniture.IsSystemBusy() == false then
    Furniture.ClearLayout()
end
```

See Also

- Furniture.ClearLayout
- Furniture.LoadLayout
- Furniture.SaveLayout

Furniture.LoadLayout

Brief

Load the user's saved furniture layout into the current scene.

Definition

```
Furniture.LoadLayout()
```

Arguments

None

Return Values

Description

Load the user's saved furniture layout into the current scene. If the call was successful, Furniture.IsSystemBusy should now return true. This function should only be called if the furniture system is not busy. This function should not be called from the furniture callback. This function should only be called if the current scene is one in which the local player is allowed to use the furniture placement system.

Examples

```
if not Furniture.IsSystemBusy() then
    Furniture.LoadLayout()
    if not Furniture.IsSystemBusy() then
        print( "Failed" )
    end
end
```

See Also

- Furniture.IsSystemBusy

- Furniture.SaveLayout

Furniture.SaveLayout

Brief

Save the furniture layout of the current scene to the user's save-data.

Definition

```
Furniture.SaveLayout()
```

Arguments

None

Return Values

Description

Save the furniture layout of the current scene to the user's save-data. If the call was successful, Furniture.IsSystemBusy should now return true. This function should only be called if the furniture system is not busy. This function should not be called from the furniture callback. This function should only be called if the current scene is one in which the local player is allowed to use the furniture placement system.

Examples

```
if not Furniture.IsSystemBusy() then
    Furniture.SaveLayout()
    if not Furniture.IsSystemBusy() then
        print( "Failed" )
    end
end
```

See Also

- Furniture.IsSystemBusy
- Furniture.LoadLayout

Furniture.SetCallback

Brief

Set the callback to receive FurnitureEvents for this script.

Definition

```
Furniture.SetCallback( function callback )
```

Arguments

callback - the callback function object

Return Values

Description

Set the callback to receive FurnitureEvents for this script. Only one callback per script.

Examples

```
function FurnitureCallback( eventId, eventData )
    if eventId == FurnitureEvent.SaveSuccess then
        mc = Furniture.SaveLayout()
    end
end
Furniture.SetCallback( FurnitureCallback )
```

See Also

- Furniture.IsSystemBusy
- Furniture.LoadLayout

GameInfo

GameInfo.GetSearchResults

Brief

Obtains the results of the search.

Definition

```
table GameInfo.GetSearchResults()
```

Arguments

None

Return Values

A table containing information on the game searched for, nil if the search was unsuccessful.

Accessible fields within the table:
title - Localized title name.
parentalLevel - Game parental level.
supportsGameLaunching - Indicates whether this game supports Game Launching explicitly.

Description

Obtains the results of the search.

Examples

```
function Update()

    if state == STATE_TITLE_SEARCH then

        if GameInfo.IsSearchFinished() == true then

            local gameInfo = GameInfo.GetSearchResults()

            if gameInfo ~= nil then

                print( string.format("Search Results:\n============\ntitle: %s\nparentalLevel:
%d\nsupportsGameLaunching: %s\n\n",
                        gameInfo.title, gameInfo.parentalLevel,
tostring(gameInfo.supportsGameLaunching) ) )

            else

                print( "GameInfo: title  not found.\n" )

            end

        end

    end

end
```

See Also

GameInfo.IsGamesListBusy

Brief

Checks to see if the games list is busy.

Definition

```
boolean GameInfo.IsGamesListBusy()
```

Arguments

None

Return Values

true if games list is busy and false otherwise.

Description

Checks to see if the games list is busy. Games list can become busy, for example, when a disc is inserted/ejected from the BD drive.

Examples

```
function Update()

    if GameInfo.IsGamesListBusy() == false then

        GameInfo.SearchByTitleId("GLMS00001")
        state = STATE_TITLE_SEARCH

    end

end
```

See Also

None

GameInfo.IsSearchFinished

Brief

Checks to see if the current search query has finished.

Definition

```
boolean GameInfo.IsSearchFinished()
```

Arguments

None

Return Values

true if search has finished, false otherwise.

Description

Checks to see if the current search query has finished.

Examples

```
function Update()

    if state == STATE_TITLE_SEARCH then

        if GameInfo.IsSearchFinished() == true then

            local gameInfo = GameInfo.GetSearchResults()

            if gameInfo ~= nil then

                print( string.format("Search Results:\n============\ntitle: %s\nparentalLevel:
%d\nsupportsGameLaunching: %s\n\n",
                        gameInfo.title, gameInfo.parentalLevel,
tostring(gameInfo.supportsGameLaunching) ) )

            else

                print( "GameInfo: title  not found.\n" )

            end

        end

    end

end
```

See Also

GameInfo.SearchByTitleId

Brief

Search for a game by title ID.

Definition

```
GameInfo.SearchByTitleId()
```

Arguments

None

Return Values

none.

Description

Search for a game by title ID. Both games installed on HDD and also currently in the BD drive will be checked.

Examples

```
function Update()

    if GameInfo.IsGamesListBusy() == false then

        GameInfo.SearchByTitleId("GLMS00001")
        state = STATE_TITLE_SEARCH

    end

end
```

See Also

None

GameLaunch

GameLaunch.CreateSession

Brief

Creates a Game Launching session and advertises it to other players in the scene.

Definition

```
GameLaunch.CreateSession()
```

Arguments

None

Return Values

None

Description

Creates a Game Launching session and advertises it to other players in the scene. Note: This should be called by the session master only.

Examples

```

```

See Also

None

GameLaunch.GetCurrentFocusPersonId

Brief

Get the personId of the person in context.

Definition

```
GameLaunch.GetCurrentFocusPersonId()
```

Arguments

None

Return Values

person ID

Description

Get the personId of the person in context.

Examples

```
function OnAddPlayer()
    local personId = GameLaunch.GetCurrentFocusPersonId();
    print( string.format("New player has joined the session. PersonId = %s", personId) );
end
```

See Also

GameLaunch.GetMaxPlayers

Brief

Retrieves the maximum number of players for this session.

Definition

```
number GameLaunch.GetMaxPlayers()
```

Arguments

None

Return Values

Maximum number of players (including session master), nil if session is invalid/not yet available.

Description

Retrieves the maximum number of players for this session. For Lua scripted GameLaunch objects, this value is set using GameLaunch.SetMaxPlayers(), For objects which use the XML menu format (specified using menu_options_xml of the game_launch object component), this value will be the user selected value from the osd item created using the <MAX_PLAYERS> tag. For the session master this value will become available once the session has been created using GameLaunch.CreateSession() For all other members, this value will become available only after successfully joining a session, indicated by the on_join_game callback.

Examples

```
function MaxPlayersOsdItemChangeCallback()
    local maxPlayers = tonumber(maxPlayersList:GetFocusItem())  -- read value from osd item
    GameLaunch.SetMaxPlayers( maxPlayers )
end
```

See Also

- GameLaunch.SetMaxPlayers

GameLaunch.GetNumberOfPlayers

Brief

Retrieves the number of players currently participating in the session.

Definition

```
number GameLaunch.GetNumberOfPlayers()
```

Arguments

None

Return Values

Number of players (including session master), nil if session is invalid/not yet available.

Description

Retrieves the number of players currently participating in the session. For the session master this value will become available once the session has been created, for all other members, this value will become available only after successfully joining a session.

Examples

```
```

See Also

None

GameLaunch.GetSessionMasterPersonId

Brief

Queries the Session Master person ID for the current session that the local player is part of.

Definition

```
GameLaunch.GetSessionMasterPersonId()
```

Arguments

None

Return Values

Session Master person ID.

Description

Queries the Session Master person ID for the current session that the local player is part of.

Examples

```
local isSessionMaster = LocalPlayer.GetMyId() == GameLaunch.GetSessionMasterPersonId();
```

See Also

None

GameLaunch.GetTitle

Brief

Retrieves information on the title used in the current Game Launch session.

Definition

```
table GameLaunch.GetTitle()
```

Arguments

None

Return Values

A table containing title information, nil if information not available.

Accessible fields within the table:
titleId - Title ID of the title.
title - Localized title name.

Description

Retrieves information on the title associated with the Game Launch session. For the Session Master it is the title chosen when creating a session. For other members it is the title that has been selected as compatible with the joining session.

Examples

```
funcation OnLocalPlayerJoinGame()
    local titleInfo = GameLaunch.GetTitle()

    if titleInfo ~= nil then
        print( string.format("Session details:\n title: %s\ntitleId: %s\n\n", titleInfo.title,
titleInfo.titleId) )

    end
end
```

See Also

None

GameLaunch.GetUserComment

Brief

Retrieves the user comment associated with the current Game Launching session.

Definition

```
string GameLaunch.GetUserComment()
```

Arguments

Return Values

The user comment set for the session (if any). nil if the session is not available.

Description

User comments are only valid once the user is part of the session.

Examples

See Also

None

GameLaunch.InitiateLaunch

Brief

Initiate the Game Launch sequence.

Definition

```
GameLaunch.InitiateLaunch()
```

Arguments

None

Return Values

None

Description

When players are ready, call this function to begin the launch sequence which includes exporting of the session options and actually launching into the title.

Examples

See Also

None

GameLaunch.LeaveSession

Brief

Submits a request to leave an existing Game Launching session.

Definition

```
GameLaunch.LeaveSession()
```

Arguments

None

Return Values

None

Description

Submits a request to leave an existing Game Launching session. This can be called by any player in a session including the session master. If a session master leaves the session will be cancelled.

Examples

See Also

- GameLaunch.CreateSession

GameLaunch.OnHideUI

Brief

Sends notification to the Game Launching system that the UI has been hidden.

Definition

```
GameLaunch.OnHideUI()
```

Arguments

None

Return Values

None

Description

Sends notification to the Game Launching system that the UI has been hidden. The session will remain active and any pad buttons reserved will be deactivated.

Examples

See Also

None

GameLaunch.OnShowUI

Brief

Sends notification to the Game Launching system that object intends to activate its UI.

Definition

```
GameLaunch.OnShowUI()
```

Arguments

None

Return Values

None

Description

Sends notification to the Game Launching system that object intends to activate its UI. Any pad buttons reserved will become activate. If the standard Game Launching Osd is used, that will be also become active. Use this function for situations when the script needs to display the object's UI and is not in response to the show_ui callback function invoked by the system. For example, the system invokes the on_launch_request callback which results in a denied launch, it maybe desirable to display a fail message using the object's UI instead of the standard pop-up Osd.

Examples

See Also

None

GameLaunch.ReturnJoinGameResult

Brief

Returns the result for the custom join game step of the joining process.

Definition

```
GameLaunch.ReturnJoinGameResult( boolean result, string localisedString, boolean showPopup )
```

Arguments

result - true = successful, false = failed.localisedString - If not nil then this string will be displayed in a popup and will replacethe standard join fail/successful message (optional argument)disablePopup - If 'localisedString' == nil then setting this to true will prevent the standard popupbeing shown. It will be the responsibility of the developer to display an appopriatemessage. (Optional argument. Default: false)

Return Values

None

Description

When a callback function in on_join_game is specified, the Game Launch system will wait for this call before proceeding. Passing 'result' as false will result in the object being immediately destroyed and all resources freed, therefore, any messages to be manually displayed should be done prior to this call.

Examples

```
localisedFailMessage = me:GetLocalisedText("fail_noEntitlement")
    GameLaunch.ReturnJoinGameResult(true, localisedFailMessage)
```

See Also

GameLaunch.ReturnLaunchRequestResult

Brief

Returns the result for the custom validation step of the launch process.

Definition

```
GameLaunch.ReturnLaunchRequestResult( boolean result, string localisedString, boolean showPopup )
```

Arguments

result - true = successful, false = failed.localisedString - If not nil then this string will be displayed in a standard Game Launching systempopup (optional argument)disablePopup - If 'localisedString' == nil then setting this to true will prevent the standard popupbeing shown. It will be the responsibility of the developer to display an appopriatemessage. (Optional argument. Default: false)

Return Values

None

Description

When a callback function in on_launch_request is specified, the Game Launch system will wait for this call before proceeding. Passing 'result' as false will result in the object being immediately destroyed and all resources freed, therefore, any messages to be manually displayed should be done prior to this call.

Examples

```
localisedFailMessage = me:GetLocalisedText("fail_notEnoughPlayers")
    GameLaunch.ReturnInitiateLaunchResult(true, localisedFailMessage)
```

See Also

GameLaunch.SetMaxPlayers

Brief

Sets the maximum number of players for the current game launching session.

Definition

```
GameLaunch.SetMaxPlayers( maxPlayers )
```

Arguments

maxPlayers - maximum number of players that can participate in a session. This value is inclusive
of the session master.

Return Values

None

Description

Sets the maximum number of players for the current game launching session. Note: This should be called by the session master only and prior to GameLaunch.CreateSession. After a session is created, subsequent calls using this function will have no effect. Note: This should be called by the session master only.

Examples

```
GameLaunch.SetMaxPlayers(12)
```

See Also

- GameLaunch.SetNumPrivateSlots

GameLaunch.SetNumPrivateSlots

Brief

Marks a set number of slots as private for the current game launching session.

Definition

```
GameLaunch.SetNumPrivateSlots( numPrivateSlots )
```

Arguments

numPrivateSlots - number of slots to be marked private. Does not include the session master.

Return Values

None

Description

Marks a set number of slots as private for the current game launching session. Note: This should be called by the session master only and must be less than the maximum number of players. The call must be made prior to GameLaunch.CreateSession. After a session is created, subsequent calls using this function will have no effect.

Examples

```
GameLaunch.SetNumPrivateSlots(6)
```

See Also

- GameLaunch.SetMaxPlayers

GameLaunch.SetUserComment

Brief

Allows the user to attach a short comment string to the Game Launching session.

Definition

```
GameLaunch.SetUserComment( string userComment )
```

Arguments

userComment - comment string

Return Values

None

Description

The comment string will be visible to other players and is designed to help them decide which session to join. Maximum comment length is 64 bytes.

Examples

```
GameLaunch.SetUserComment("Noobs only")
```

See Also

None

GameLaunchButton

GameLaunchButton.GetNameRef

Brief

Obtains a button's reference name.

Definition

```
string GameLaunchButton.GetNameRef(GameLaunchButton buttonItem)
```

Arguments

Return Values

Returns the name reference string of the button item passed in.

Description

For buttons which contain localized text, the reference string is returned. For buttons non localized text, the non-localized string is returned. GetNameRef() can be useful for debug purposes and also in cases where a developer wishes to manually write out a custom session XML file upon launching. Note: Contents of GameLaunchButton types are not exported to an GAMELAUNCHSESSION.XML

Examples

```
local item = myMenu1:GetFirstItem()

    while item ~= nil do

        if Type(item) == "GameLaunchButton" then
            local buttonName = item:GetNameRef()

            if buttonName == "myButtonNameRef" then

                GameLaunchButton.SetText( item, "My Localized String", false )

            end
        end

        item = myMenu1:GetNextItem(item)
    end
```

See Also

GameLaunchButton.SetCallbackOnPressed

Brief

Specifies a callback function to executed when the button is pressed.

Definition

```
GameLaunchButton.SetCallbackOnPressed(GameLaunchButton button, string callbackFunction)
```

Arguments

Return Values

Description

Specifies a callback function to executed when the button is pressed.

Examples

```
myButton = mainMenu:AddItemButton( "REF_MyUserButton" );
GameLaunchButton.SetCallbackOnPressed( myButton, "MyButtonCB()" )

function MyButtonCB()

    GameLaunchButton( myButton, "REF_buttonPressed", true )

end
```

See Also

GameLaunchButton.SetText

Brief

Sets the text displayed in the button.

Definition

```
GameLaunchButton.SetText(GameLaunchButton button, string buttonText, boolean needsLocalization)
```

Arguments

button - This button object.buttonText - The text which should be displayed in the button.needsLocalization - Specifies whether 'buttonText' should be localized before it is displayed. This is
an optional parameter and defaults to true if unspecified.

Return Values

Description

Sets the text displayed in the button. If the parameter 'needsLocalization' is true, this button's refernce string will become the value of 'buttonText' and will be the string returned by GameLaunchButton.GetNameRef()

Examples

```
myButton = mainMenu:AddItemButton( "REF_MyUserButton" );
GameLaunchButton.SetCallbackOnPressed( myButton, "MyButtonCB()" )

function MyButtonCB()

    GameLaunchButton( myButton, "REF_buttonPressed", true )

end
```

See Also

GameLaunchExporter

GameLaunchExporter.BeginSection

Brief

Begins a new section with the name specified.

Definition

```
GameLaunchExporter.BeginSection()
```

Arguments

Section name

Return Values

None

Description

Begins a new section with the name specified. Must be called after BeginSessionExport and EndSessionExport, and must be accompanied by an EndSection.

Examples

```
GameLaunchExporter.BeginSessionExport()
    GameLaunchExporter.BeginSection("sessionOptions")
        GameLaunchExporter.ExportOption("Track", "TrackA")
    GameLaunchExporter.EndSection()
GameLaunchExporter.EndSessionExport()
```

See Also

- GameLaunchExporter.EndSessionExport
- GameLaunchExporter.EndSection

GameLaunchExporter.BeginSessionExport

Brief

Begins the export of the GAMELAUNCHSESSION.XML file

Definition

```
GameLaunchExporter.BeginSessionExport()
```

Arguments

None

Return Values

None

Description

Begins the export of the GAMELAUNCHSESSION.XML file. Call GameLaunchExporter.EndSessionExport() to finish exporting.

Examples

```
GameLaunchExporter.BeginSessionExport()
    -- your Sections here ...
GameLaunchExporter.EndSessionExport()
```

See Also

- GameLaunchExporter.EndSessionExport

GameLaunchExporter.EndSection

Brief

Ends a section that was began with BeginSection

Definition

```
GameLaunchExporter.EndSection()
```

Arguments

None

Return Values

None

Description

Ends a section that was began with BeginSection

Examples

```
GameLaunchExporter.BeginSessionExport()
    GameLaunchExporter.BeginSection("sessionOptions")
        GameLaunchExporter.ExportOption("Track", "TrackA")
    GameLaunchExporter.EndSection()
GameLaunchExporter.EndSessionExport()
```

See Also

- GameLaunchExporter.BeginSessionExport
- GameLaunchExporter.BeginSection

GameLaunchExporter.EndSessionExport

Brief

Finalizes and writes out the GAMELAUNCHSESSION.XML file

Definition

```
GameLaunchExporter.EndSessionExport()
```

Arguments

None

Return Values

None

Description

Finalizes and writes out the GAMELAUNCHSESSION.XML file

Examples

```
GameLaunchExporter.BeginSessionExport()
    -- your Sections here ...
GameLaunchExporter.EndSessionExport()
```

See Also

- GameLaunchExporter.BeginSessionExport

GameLaunchExporter.ExportOption

Brief

Exports nodes item and value based on two passed string values.

Definition

```
GameLaunchExporter.ExportOption()
```

Arguments

String item, String value

Return Values

None

Description

Exports nodes item and value based on two passed string values.

Examples

```
GameLaunchExporter.BeginSection("sessionOptions")
    GameLaunchExporter.ExportOption("Track", "TrackA")
    GameLaunchExporter.ExportOption("Ambiance", "1")
GameLaunchExporter.EndSection()
```

See Also

- GameLaunchExporter.BeginSessionExport

GameLaunchListItem

GameLaunchListItem.AddItem

Brief

Adds a item to a GameLaunchListItem menu item.

Definition

```
string GameLaunchListItem.AddItem(GameLaunchListItem listItem, string itemNameRef)
```

Arguments

listItem - The list item object to which the item is to be added.
itemNameRef - The localisation reference name of the item to be added.

Return Values

Description

Items added via AddItem() must have an entry in the object's localisation database. If raw values are desired which are not subject to any localisation then AddItemRaw() can be used.

Examples

```
trackList = myMenu1:AddItemList("track")
trackList:AddItem("trackA")
trackList:AddItem("trackB")
trackList:AddItem("trackC")
```

See Also

None

GameLaunchListItem.AddItemRaw

Brief

Adds a item to a GameLaunchListItem menu item.

Definition

```
string GameLaunchListItem.AddItemRaw(GameLaunchListItem listItem, string itemRawText)
```

Arguments

listItem - The list item object to which the item is to be added.
itemRawText - The actual text string for the item to be added.

Return Values

Description

When using AddItemRaw() the string specified by itemRawText will be directly displayed without localisation. If localised strings are required AddItem() should be used.

Examples

```
maxPlayersList = myMenu1:AddItemList("maxPlayers")
maxPlayersList:AddItemRaw("2")
maxPlayersList:AddItemRaw("4")
maxPlayersList:AddItemRaw("6")
```

See Also

None

GameLaunchListItem.GetCallbackFocusItem

Brief

Gets the GameLaunchListItem object for which a focus item has changed.

Definition

```
GameLaunchListItem GameLaunchListItem.GetCallbackFocusItem()
```

Arguments

Return Values

The GameLaunchListItem object for which a focus item has changed.

Description

This function is valid when used within a callback function that was registered with GameLaunchListItem.SetCallbackOnChange()

Examples

```
function OnSessionSetup()
    maxPlayersList = myMenu1:AddItemList("maxPlayers")
    maxPlayersList:AddItemRaw("2")
    maxPlayersList:AddItemRaw("4")
    maxPlayersList:AddItemRaw("6")
    maxPlayersList:SetCallbackOnChange("MaxPlayersChangeCB()")
end

function MaxPlayersChangeCB()
    local listItem = GameLaunchListItem.GetCallbackFocusItem()

    maxPlayers = tonumber( listItem:GetFocusItem() )
end
```

See Also

None

GameLaunchListItem.GetFocusItem

Brief

Queries the current item under focus.

Definition

```
string GameLaunchListItem.GetFocusItem(GameLaunchListItem listItem)
```

Arguments

listItem - The GameLaunchListItem object on which the query is performed.

Return Values

If the item was added using AddItem(), the localisation reference string is returned.
If the item was added using AddItemRaw(), the raw string will be returned.

Description

Examples

```
local listItem = myMenu:GetItemByNameRef("numTracks")

numTracks = tonumber( listItem:GetFocusItem() )
```

See Also

None

GameLaunchListItem.GetNameRef

Brief

Obtains a list item's reference name.

Definition

```
string GameLaunchListItem.GetNameRef(GameLaunchListItem listItem)
```

Arguments

Return Values

Returns the name reference string of the list item item passed in.

Description

GetNameRef() can be useful for debug purposes and also in cases where a developer wishes to manually write out a custom session XML file upon launching.

Examples

```
local item = myMenu1:GetFirstItem()

while item ~= nil do
    -- Reset all sliders

    if Type(item) == "GameLaunchListItem" then
        local listName = item:GetNameRef()

        -- Insert code here.

    end

    item = myMenu1:GetNextItem(item)
end
```

See Also

None

GameLaunchListItem.RemoveAllItems

Brief

Removes all the items from a GameLaunchListItem menu item.

Definition

```
GameLaunchListItem.RemoveAllItems(GameLaunchListItem listItem)
```

Arguments

listItem - The GameLaunchListItem object

Return Values

Description

Examples

```
local myList = myMenu1:FindItemByNameRef("stage3")
myList:RemoveAllItems()
```

See Also

None

GameLaunchListItem.RemoveItem

Brief

Removes an item from a GameLaunchListItem menu item.

Definition

```
GameLaunchListItem.RemoveItem(GameLaunchListItem listItem, string itemNameRef)
```

Arguments

listItem - The GameLaunchListItem object
itemNameRef - The reference string of the item to be removed. If the item was added using AddItem() then the localisation reference string should be passed in, if AddItemRaw() was used then pass in the raw string instead.

Return Values

Description

Examples

```
local myList = myMenu1:FindItemByNameRef("stage3")
myList:RemoveItem("option2")
```

See Also

None

GameLaunchListItem.SetCallbackOnChange

Brief

Sets a callback function to be called when an item selection changes.

Definition

```
GameLaunchListItem.SetCallbackOnChange(string callbackFunction)
```

Arguments

callbackFunction - The name of the callback function. The exact string of a function call should be specified including parenthesis.

Return Values

Description

Examples

```
function OnSessionSetup()
    maxPlayersList = myMenu1:AddItemList("maxPlayers")
    maxPlayersList:AddItemRaw("2")
    maxPlayersList:AddItemRaw("4")
    maxPlayersList:AddItemRaw("6")
    maxPlayersList:SetCallbackOnChange("MaxPlayersChangeCB()")
end

function MaxPlayersChangeCB()
    local listItem = GameLaunchListItem.GetCallbackFocusItem()

    maxPlayers = tonumber( listItem:GetFocusItem() )
end
```

See Also

None

GameLaunchListItem.SetFocusItem

Brief

Sets the current item under focus.

Definition

```
GameLaunchListItem.SetFocusItem(GameLaunchListItem listItem, string itemNameRef)
```

Arguments

listItem - The GameLaunchListItem object
itemNameRef - The reference string of the item that is to become the focus item. If the item was added using AddItem() then the localisation reference string should be passed in, if AddItemRaw() was used then just the raw string instead.

Return Values

Description

Examples

```
maxPlayersList = myMenu1:AddItemList("maxPlayers")
maxPlayersList:AddItemRaw("2")
maxPlayersList:AddItemRaw("4")
maxPlayersList:AddItemRaw("8")
maxPlayersList:SetFocusItem("8")
```

See Also

None

GameLaunchMenu

GameLaunchMenu.AddItemButton

Brief

Adds a button item to the menu specified.

Definition

```
GameLaunchButton GameLaunchMenu.AddItemButton(GameLaunchMenu menu, string itemNameRef, string
insertRef)
```

Arguments

menu - Menu to where the button should be added to.itemNameRef - Localization reference string for the display name of this button.insertRef - Specifies the position of where the item should be inserted. The 'itemNameRef' of an existing item can be passed. This is an optional argument, not specifying
a value will result in the button being appended as the last item.

Return Values

The newly added GameLaunchButton instance.

Description

Adds a button item to the menu specified.

Examples

```
mainMenu = GameLaunchMenuSys.CreateMenu("mainMenu", "mainMenu", true)

if mainMenu ~= nil then

    local userComments = me:GetLocalisedText("ref_userComments")
    myButton = myMenu1:AddItemButton(userComments)

end
```

See Also

GameLaunchMenu.AddItemList

Brief

Adds a list item to the menu specified.

Definition

```
GameLaunchListItem GameLaunchMenu.AddItemList(GameLaunchMenu menu, string listItemNameRef)
```

Arguments

menu - Menu to where the list should be added to.itemNameRef - Localization reference string for the display name of this slider.insertPosition - Specifies the position of where the item should be inserted. The 'itemNameRef' of an existing item can be passed. This is an optional argument, not specifying
a value will result in the slider being appended as the last item.

Return Values

The newly added GameLaunchListItem instance.

Description

Adds a list item to the menu specified.

Examples

```
mainMenu = GameLaunchMenuSys.CreateMenu("mainMenu", "mainMenu", true)

if mainMenu ~= nil then

    spawnDelayList = mainMenu:AddItemList("spawnDelay")
    spawnDelayList:AddItem("none")
    spawnDelayList:AddItem("oneSecond")
    spawnDelayList:AddItem("twoSeconds")
    spawnDelayList:AddItem("threeSeconds")
    spawnDelayList:AddItem("sixSeconds")
    spawnDelayList:SetFocusItem("none")

end
```

See Also

GameLaunchMenu.AddItemSlider

Brief

Adds a slider item to the menu specified.

Definition

```
GameLaunchSlider GameLaunchMenu.AddItemSlider(GameLaunchMenu menu, string itemNameRef, number
minValue, number maxValue, number stepValue, number defaultValue, string insertRef)
```

Arguments

menu - Menu to where the slider should be added to.itemNameRef - Localization reference string for the display name of this slider.minValue - The slider's minimum value.maxValue - The slider's maximum value.stepValue - The step value.defaultValue - Default valueinsertRef - Specifies the position of where the item should be inserted. The 'itemNameRef' of an existing item can be passed. This is an optional argument, not specifying
a value will result in the slider being appended as the last item.

Return Values

The newly added GameLaunchSlider instance.

Description

Adds a slider item to the menu specified.

Examples

```
mainMenu = GameLaunchMenuSys.CreateMenu("mainMenu", "mainMenu", true)

if mainMenu ~= nil then

    mySlider1 = myMenu1:AddItemSlider("difficulty", 1, 8, 1, 1)
    mySlider1:SetPercentageSlider(false)
    mySlider1:SetIntegerSlider(true)

end
```

See Also

GameLaunchMenu.FindItemByNameRef

Brief

Finds a menu item within a given menu.

Definition

```
GameLaunchItem GameLaunchMenu.FindItemByNameRef(GameLaunchMenu menu, string itemNameRef)
```

Arguments

menu - The menu to query.itemNameRef - The localization reference string of the item being searched for.

Return Values

If the search was successful the object of the item is returned. Valid return types are GameLaunchItemList
and GameLaunchSlider. If the search failed nil will be returned.

Description

Finds a menu item within a given menu.

Examples

```
local brightnessSlider = menu1:FindItemByNameRef("brightness")

if brightnessSlider ~= nil then
    brightnessSlider:SetValue("20")
end
```

See Also

GameLaunchMenu.GetFirstItem

Brief

Returns the first menu item in a given menu.

Definition

```
[GameLaunchItemList | GameLaunchSlider] GameLaunchMenu.GetFirstItem(GameLaunchMenu menu)
```

Arguments

menu - The menu to query.

Return Values

If successful the object of the item is returned. Valid return types are GameLaunchItemList and GameLaunchSlider. If the search failed nil will be returned.

Description

To be used in conjunction with GameLaunchMenu.GetNextItem()

Examples

```
local item = myMenu1:GetFirstItem()

while item ~= nil do
    -- Reset all sliders

    if Type(item) == "GameLaunchSlider" then
        item:SetValue("0")
    end

    item = myMenu1:GetNextItem(item)
end
```

See Also

GameLaunchMenu.GetId

Brief

Obtains a menu's Id string.

Definition

```
string GameLaunchMenu.GetId(GameLaunchMenu menu)
```

Arguments

menu - The menu to query.

Return Values

Returns the Id of the menu passed in.

Description

Obtains a menu's Id string.

Examples

```
local menu1 = GameLaunchMenuSys.GetFirstMenu()

while menu1 ~= nil do

    if menu1:GetId() == "settings" then

        local brightnessSlider = menu1:FindItemByNameRef("brightness")

        if brightnessSlider ~= nil then
            brightnessSlider:SetValue("20")
        end

        break
    end

    menu1 = GameLaunchMenuSys.GetNextMenu(menu1)
end
```

See Also

GameLaunchMenu.GetNextItem

Brief

Returns the next menu item in a given menu.

Definition

```
[GameLaunchItemList | GameLaunchSlider] GameLaunchMenu.GetNextItem(GameLaunchMenu menu,
[GameLaunchItemSlider|GameLaunchSlider] item)
```

Arguments

menu - The menu to query.item - The item return from the previous call to GetFirstItem() or GetNextItem(), whichever was the most recent.

Return Values

If successful the object of the item is returned. Valid return types are GameLaunchItemList
and GameLaunchSlider. If the search failed nil will be returned.

Description

To be used in conjunction with GameLaunchMenu.GetFirstItem()

Examples

```
local item = myMenu1:GetFirstItem()

while item ~= nil do
    -- Reset all sliders

    if Type(item) == "GameLaunchSlider" then
        item:SetValue("0")
    end

    item = myMenu1:GetNextItem(item)
end
```

See Also

GameLaunchMenuSys

GameLaunchMenuSys.AddMenu

Brief

Adds an menu screen to the menu system.

Definition

```
GameLaunchMenuSys.AddMenu( GameLaunchMenu menu, string insertPosition )
```

Arguments

menu - GameLaunchMenu object return from GameLaunchMenuSys.CreateMenu()insertPosition - specifies the position where the menu should be inserted. The menu will be inserted before the
menu with the Id given. This is an optional argument, if it is not specified the menu will be appended to the end.

Return Values

None.

Description

Adds an menu screen to the menu system.

Examples

```
myMenu1 = GameLaunchMenuSys.CreateMenu( "myMenu1", "mainMenu", true )

friendlyFireItem = GameLaunchMenu.AddItemList( myMenu1, "friendlyFire" )
GameLaunchListItem.AddItem( friendlyFireItem, "on" )
GameLaunchListItem.AddItem( friendlyFireItem, "off" )

GameLaunchMenuSys.AddMenu( myMenu1 )
```

See Also

None

GameLaunchMenuSys.CreateMenu

Brief

Creates and returns a GameLaunchMenu object.

Definition

```
GameLaunchMenu GameLaunchMenuSys.CreateMenu( string menuId, string menuNameRef, boolean isSessionMenu
)
```

Arguments

menuId - A unique Id string used to identify this menu screen and is used when exporting the session XML file.menuNameRef - Name given to the menu which will be displayed on screen. Reference string used to look up the translated string in the object's localisation database.isSessionMenu - Session menus are controlled by the Session Master, options contained within such menus become read only once the session has been created.

Return Values

GameLaunchMenu

Description

Creates and returns a GameLaunchMenu object.

Examples

```
menu1 = GameLaunchMenuSys.CreateMenu( "myMenu1", "mainMenu", true )
```

See Also

None

GameLaunchMenuSys.FindMenuById

Brief

Finds a menu from within the menu system.

Definition

```
GameLaunchMenu GameLaunchMenuSys.FindMenuById( string menuId )
```

Arguments

menuId - Menu Id of the menu to search for.

Return Values

GameLaunchMenu object, or nil if not found.

Description

Finds a menu from within the menu system.

Examples

```
local menu = GameLaunchMenuSys.FindMenuById( "myMenu3" )

if menu ~= nil then
    local mySlider1 = GameLaunchMenu.AddItemSlider( menu, "difficulty", 1, 10, 1, 3 )
    GameLaunchSlider.SetPercentageSlider( mySlider1, false )
    GameLaunchSlider.SetIntegerSlider( mySlider1, true )
end
```

See Also

None

GameLaunchMenuSys.GetFirstMenu

Brief

Retrieves the first menu from the menu system.

Definition

```
GameLaunchMenu GameLaunchMenuSys.GetFirstMenu()
```

Arguments

None

Return Values

GameLaunchMenu object, or nil if none found.

Description

Retrieves the first menu from the menu system. Useful when searching for active menus where the GameLaunchMenu object isn't already known, for example, menus created by the Game Launching XML menu format.

Examples

```
local myMenu = GameLaunchMenuSys.GetFirstMenu()

if myMenu ~= nil then

    ...

end
```

See Also

None

GameLaunchMenuSys.GetNextMenu

Brief

Retrieves the next menu from the menu system.

Definition

```
GameLaunchMenu GameLaunchMenuSys.GetNextMenu( GameLaunchMenu myMenu )
```

Arguments

myMenu - GameLaunchMenu object return from the previous GetFirstMenu() or GetNextMenu() call, whichever was the most recent.

Return Values

GameLaunchMenu object, or nil if none found.

Description

Retrieves the next menu from the menu system. To be used in conjunction with GameLaunchMenuSys.GetFirstMenu() Useful when searching for active menus where the GameLaunchMenu object isn't already known, for example, menus created by the Game Launching XML menu format.

Examples

```
local aMenu = GameLaunchMenuSys.GetFirstMenu()

if aMenu ~= nil then

    numMenus = 1

    repeat

        aMenu = GameLaunchMenuSys.GetNextMenu(aMenu)

        if aMenu ~= nil then
            numMenus = numMenus + 1
        end

    until aMenu == nil

end
```

See Also

- GameLaunchMenuSys.GetFirstMenu
- GameLaunchMenuSys.GetNextMenu

GameLaunchMenuSys.RemoveMenu

Brief

Removes an active menu from the menu system.

Definition

```
GameLaunchMenuSys.RemoveMenu( string menuId )
```

Arguments

menuId - Id of the menu to be removed.

Return Values

None.

Description

Removes an active menu from the menu system. This does not destroy the object instance. If the destruction of the object instance is required, this should be done via Lua garbage collection.

Examples

```
local myMenu = GameLaunchMenuSys.FindMenuById( "myMenu1" )

if myMenu ~= nil then
    GameLaunchMenuSys.RemoveMenu( "myMenu" )
end
```

See Also

None

GameLaunchSlider

GameLaunchSlider.GetNameRef

Brief

Obtains a slider's reference name.

Definition

```
string GameLaunchSlider.GetNameRef(GameLaunchSlider sliderItem)
```

Arguments

Return Values

Returns the name reference string of the slider item passed in.

Description

GetNameRef() can be useful for debug purposes and also in cases where a developer wishes to manually write out a custom session XML file upon launching.

Examples

```
local item = myMenu1:GetFirstItem()

while item ~= nil do
    -- Reset all sliders

    if Type(item) == "GameLaunchSlider" then
        local sliderName = item:GetNameRef()

        -- Insert code here.

    end

    item = myMenu1:GetNextItem(item)
end
```

See Also

None

GameLaunchSlider.GetValue

Brief

Gets the current value of the slider.

Definition

```
GameLaunchSlider.GetValue(GameLaunchSlider sliderItem, number value)
```

Arguments

sliderItem - The slider object

Return Values

Description

Examples

```
local levelSlider = myMenu1:FindItemByNameRef("level")

if levelSlider ~= nil then
    currentLevel = levelSlider:GetValue()
end
```

See Also

None

GameLaunchSlider.SetIntegerSlider

Brief

Toggles the slider as a integer type slider.

Definition

```
GameLaunchSlider.SetIntegerSlider(GameLaunchSlider sliderItem, boolean enable)
```

Arguments

sliderItem - The slider object
enable - Flag to specify if slider should be a integer type slider. If this value is false, the slider will use floating point values instead.

Return Values

Description

Examples

```
levelSlider = myMenu1:AddItemSlider("level", 1, 10, 1, 3)
levelSlider:SetPercentageSlider(false)
levelSlider:SetIntegerSlider(true)
```

See Also

None

GameLaunchSlider.SetPercentageSlider

Brief

Toggles the slider as a percentage type slider.

Definition

```
GameLaunchSlider.SetPercentageSlider(GameLaunchSlider sliderItem, boolean enable)
```

Arguments

sliderItem - The slider object
enable - Flag to specify if slider should be a percentage type slider.

Return Values

Description

Note: Percentage sliders using floating point values is not supported.

Examples

```
energySlider = myMenu1:AddItemSlider("energy", 1, 100, 10, 80)
energySlider:SetPercentageSlider(true)
energySlider:SetIntegerSlider(true)
```

See Also

None

GameLaunchSlider.SetValue

Brief

Sets the value of the slider.

Definition

```
GameLaunchSlider.SetValue(GameLaunchSlider sliderItem, number value)
```

Arguments

sliderItem - The slider object
value - The value to which the slider should be set to.

Return Values

Description

Examples

```
local slider = myMenu1:FindItemByNameRef("level")

if slider ~= nil then
    slider:SetValue(3)
end
```

See Also

None

GfxEffect

GfxEffect.Create

Brief

Creates a new GfxEffect object.

Definition

```
GfxEffect GfxEffect.Create( )
```

Arguments

None

Return Values

A new GfxEffect instance.

Description

Creates a new GfxEffect object. By default, an effect is not enabled and will not render. To enable the effect, call GfxEffect.Enable.

Examples

```
gfxEffect = GfxEffect.Create()
-- set up geometry
gfxEffect:Enable(true)
```

See Also

- GfxEffect.Enable

GfxEffect.Enable

Brief

Enable or disable rendering of the GfxEffect.

Definition

```
GfxEffect.Enable(GfxEffect effect, boolean enable)
```

Arguments

effect - Effect to modify.enable - True to enable or false to disable.

Return Values

None.

Description

This function controls whether the GfxEffect instance is rendered or not. This state has no effect on the data within the effect, so it can be enabled and disabled without re-specifying the data. If you wish to delete the effect altogether, allow it to be garbage collected.

Examples

```
gfxEffect:Enable(false)
```

See Also

- GfxEffect.IsEnabled

GfxEffect.IsEnabled

Brief

Indicates whether the GfxEffect instance is enabled.

Definition

```
boolean GfxEffect.IsEnabled(GfxEffect effect)
```

Arguments

effect - Effect to query.

Return Values

True if enabled or false otherwise.

Description

An effect must be enabled in order to render.

Examples

```
if (not gfxEffect:IsEnabled()) then
    gfxEffect:Enable(true)
end
```

See Also

- GfxEffect.Enable

GfxEffect.SetAlphaRef

Brief

Set the reference value used for cutout effects.

Definition

```
GfxEffect.SetAlphaRef( GfxEffect effect, number alphaRef )
```

Arguments

effect - The GfxEffect instance.alphaRef - The reference value to use for alpha testing.

Return Values

None.

Description

An effect rendered with the cutout blend mode will apply an alpha test when rendering, using the value supplied to this function as the reference value. Any geometry with an alpha value greater than or equal to the reference value will be rendered, alpha values lower than the reference will not be rendered.

Examples

```
```

See Also

- GfxEffect.SetBlendMode

GfxEffect.SetBlendMode

Brief

Set the blending mode for the effect.

Definition

```
GfxEffect.SetBlendMode( GfxEffect effect, BlendMode mode )
```

Arguments

effect - The effect instance.mode - The desired blend mode. Valid values are:
BlendMode.Opaque => no blend, render opaque
BlendMode.Cutout => render opaque with alpha test using AlphaRef value
BlendMode.Blend => SrcAlpha*Src + (1-SrcAlpha)*Dst
BlendMode.Add => Src + Dst
BlendMode.Multiply => Src * Dst
BlendMode.AlphaAdd => SrcAlpha*Src + Dst
BlendMode.BlendPremultiplied => Src + (1-SrcAlpha)*Dst

Return Values

None.

Description

The blend mode controls how the GfxEffect is blended onto the final frame. Opaque renders the geometry unblended, and cutout applies an alpha test with the threshold set with SetAlphaRef when rendering. For effect of other modes please refer to BlendMode value list in this help entry.

Examples

See Also

- GfxEffect.SetAlphaRef
- GfxEffect.SetRenderStage
- GfxEffect.SetDepthPrepass

GfxEffect.SetBounds

Brief

Sets the local axis aligned bounding box of the effect.

Definition

```
GfxEffect.SetBounds( GfxEffect effect, Vector4 min, Vector4 max )
```

Arguments

effect - The instance to modify.min - The local space minimum extent of the effect.max - The local space maximum extent of the effect.

Return Values

None.

Description

GfxEffect instances require a local bounding box in order to be culled when not in view. The bounding box should encompass all of the geometry created by the effect.

Examples

```
local vertData = MemoryContainer.Create(numVerts * 4)       -- 4 bytes per vertex
local offset = 0
for i = 1, numVerts do
    local v = GenerateVert(i)       -- user function to compute some vertex position
    vertData:SetVector4(offset, v)
    offset = offset + 4
end
-- compute bounds
local min = Vector4.Create(Const.MaxNum, Const.MaxNum, Const.MaxNum)
local max = -min
StreamOps.BoundsVec4(verts, 0, numVerts, min, max)

-- set up gfxeffect
gfxEffect = GfxEffect.Create()
gfxEffect:SetVertexData(GfxVertAttrib.Position, vertData, 4)
-- will also need some indices here...
gfxEffect:SetBounds(min, max)
```

See Also

- StreamOps.BoundsVec4

GfxEffect.SetDepthPrepass

Brief

Set whether the effect renders a depth prepass.

Definition

```
GfxEffect.SetDepthPrepass( GfxEffect effect, boolean flag )
```

Arguments

effect - The GfxEffect instance.flag - True to turn on depth prepass rendering, false otherwise.

Return Values

None.

Description

Setting this flag will render the effect into the depth buffer first, and only render faces passing an equal depth test when rendering to the back buffer. Defaults to off.

Examples

```

```

See Also

- GfxEffect.SetBlendMode

GfxEffect.SetDrawRange

Brief

Sets the range of triangles to draw.

Definition

```
GfxEffect.SetDrawRange( GfxEffect effect, number start, number count )
```

Arguments

effect - The instance to set the draw range on.start - Offset into indices at which to start drawing.count - The number of triangles to draw.

Return Values

None.

Description

It is more optimal to create one GfxEffect and render subsections of its geometry than create separate GfxEffects. This function can be used to draw a subset of the triangles in the GfxEffect.

Examples

```
gfxEffect:SetDrawRange(6, 12)   -- draw 12 triangles starting from index 6 (ie, skip first 2 tris)
```

See Also

- GfxEffect.SetIndexData

GfxEffect.SetIndexData

Brief

Sets the triangle indices for the effect.

Definition

```
GfxEffect.SetIndexData( GfxEffect effect, MemoryContainer indices, number offset = 0 )
```

Arguments

effect - The instance to set the indices on.indices - The memory container with vertex indices as a triangle list, 3 indices per triangle.
The memory container should be allocated in host memory and 128-byte aligned.offset - The offset into the memory container at which to find the indices.

Return Values

None.

Description

The geometry is specified as a triangle list with 3 indices per triangle. As such there will be triangleCount * 3 indices required. Each index is assumed to be 2 bytes in size (65k vertex limit). Note that the index data is required to be 128-byte aligned, and located in host memory. The memory is used by the GfxEffect directly, i.e. no copy is made, so the user should ensure the memory container exists for as long as the GfxEffect exists. Clockwise winding indicates a front-facing triangle. The number of indices to use is indicated by the SetDrawRange function.

Examples

```

```

See Also

- GfxEffect.SetVertexData
- GfxEffect.SetDrawRange

GfxEffect.SetRenderStage

Brief

Set the render stage for semi-transparent effects.

Definition

```
GfxEffect.SetRenderStage( GfxEffect effect, RenderStage mode )
```

Arguments

effect - The effect instance.mode - The desired render stage. Valid values are:
RenderStage.PreWorld
RenderStage.PostWorld
RenderStage.PostParticles

Return Values

None.

Description

For effects being blended into the frame, this function controls the render stage at which the effect is rendered. Geometry rendered in subsequent stages will be blended on top of geometry rendered at at earlier stage. The render stages are listed in this entry in temporal order.

Examples

See Also

GfxEffect.SetShadowEnable

Brief

Set whether the effect casts a dynamic shadow.

Definition

```
GfxEffect.SetShadowEnable( GfxEffect effect, boolean flag )
```

Arguments

effect - The GfxEffect instance.flag - True to cast a dynamic shadow from this effect, or false otherwise.

Return Values

None.

Description

Enabling this flag will cause the effect to cast a dynamic shadow onto the scene, at a small performance cost. Defaults to off.

Examples

See Also

None

GfxEffect.SetTexture

Brief

Sets the given texture on the given shader uniform parameter.

Definition

```
GfxEffect.SetTexture( GfxEffect effect, TextureType type, Texture|string|resource tex )
```

Arguments

effect - The GfxEffect instance.type - The texture type to set:
TextureType.ColorMap
tex - The texture, texture resource name, or texture resource.

Return Values

None.

Description

Set the texture to use for the specified texture slot.

Examples

```
tex = Texture.Find("water")
gfxEffect:SetTexture( TextureType.ColorMap, tex )
```

See Also

- Texture.Find
- Resource.Find

GfxEffect.SetTransform

Brief

Sets the world transform for the effect.

Definition

```
GfxEffect.SetTransform( GfxEffect effect, Matrix44 transform )
```

Arguments

effect - The instance to modify.transform - The world space tranform.

Return Values

None.

Description

Sets the world transform for the effect, used to place it within the scene.

Examples

```
local transform = Matrix44.Create()
transform:SetTranslation(Vector4.Create(1, 2, 3))
gfxEffect:SetTransform(transform)
```

See Also

None

GfxEffect.SetTwoSided

Brief

Set whether the effect is rendered two-sided.

Definition

```
GfxEffect.SetTwoSided( GfxEffect effect, boolean flag )
```

Arguments

effect - The GfxEffect instance.flag - True to render two-sided, false otherwise.

Return Values

None.

Description

If the geometry should be considered two-sided, set this flag to true. Defaults to on.

Examples

See Also

None

GfxEffect.SetVertexData

Brief

Sets a vertex data attribute for the effect.

Definition

```
boolean GfxEffect.SetVertexData( GfxEffect effect, VertexDataType type, MemoryContainer data, number
numElementsPerVertex, number offset = 0, number mode = DataRepeat.None, number frequency = 0)
```

Arguments

effect - The instance to set the vertex data into.type - Vertex data type:
VertexDataType.InPosition
VertexDataType.InTexCoord1
VertexDataType.InColordata - Memory container specifying the vertex data. The memory container should be allocated in host memory.numElementsPerVertex - number of elements, range is [1, 4].mode - The desired vertex data repeat mode. Valid values are:
DataRepeat.None
DataRepeat.Element
DataRepeat.Bufferfrequency - the data repeat frequency, a value of 0 disables element or buffer repeat.

Return Values

True if the call succeeded, false otherwise.

Description

Supply the vertex data (position, colour, texture coords) for a GfxEffect. Note that the memory is used by the GfxEffect directly, no copy is made. Therefore the user should ensure the memory container exists for as long as the GfxEffect exists. All elements are floats, including color. All vertex data streams must be specified for the effect to render.

Examples

See Also

- GfxEffect.SetIndexData

Group

Group.Create

Brief

Create a group.

Definition

```
boolean Group.Create()
```

Arguments

None

Return Values

True if the group creation process started, false otherwise (already a member of a group).

Description

Triggers the group creation process. A script should query for group membership to determine when the group has been created. This function will fail if the user is already a member of a group.

Examples

```
if (Group.Create()) then
    state = CREATING_GROUP
end
```

See Also

- Group.Leave
- Group.GetId
- Group.IsInGroup
- Group.Join

Group.GetId

Brief

Get the ID of the currently joined group.

Definition

```
boolean Group.GetId(BigInt<64> id)
```

Arguments

id - 64-bit BigInt to receive the group ID.

Return Values

True upon success, false if not a member of a group.

Description

Obtain the ID of the group which the current user belongs to. If they are not a member of a group this function will return false and the BigInt is not

modified.

Examples

```
local groupId = BigInt.Create(64)
if (Group.GetId(groupId)) then
    local msg = OutboundMessage.Create(JOIN_GROUP, 8)
    msg:WriteBigInt(groupId)
    msg:Send()
end
```

See Also

- Group.Join
- Group.Create

Group.GetMemberCount

Brief

Get the number of members of the current user's group.

Definition

```
number Group.GetMemberCount()
```

Arguments

None

Return Values

The number of group members.

Description

Gets the number of members of the current user's group. If the current user does not have a group, this function will raise an error.

Examples

```
local numMembers = Group.GetMemberCount()
```

See Also

- Group.GetMemberList
- Group.IsInGroup

Group.GetMemberList

Brief

Get the list of members of the current user's group.

Definition

```
table Group.GetMemberList()
```

Arguments

None

Return Values

An array with one table for each member, with the following fields:

{
number id,
User user
}

Description

Get the list of members of the current user's group. If the current user does not have a group, this function will raise an error.

Examples

```
local members = Group.GetMemberList()
print('Group members:')
for i, m in ipairs(members) do
    print(m.user:GetName())
end
```

See Also

- Group.GetMemberCount
- Group.IsInGroup

Group.IsInGroup

Brief

Returns true if the current member is in a group.

Definition

```
boolean Group.IsInGroup()
```

Arguments

None

Return Values

true if the current user is in a group, false otherwise.

Description

Returns true if the current member is in a group.

Examples

```
if (Group.IsInGroup()) then
    print('You have ' .. Group.GetGroupMemberCount() .. ' people in your group!')
end
```

See Also

- Group.GetMemberCount
- Group.GetMemberList

Group.Join

Brief

Join an existing group.

Definition

```
boolean Group.Join(BigInt<64> groupId)
```

Arguments

None

Return Values

True if the group join process started, false otherwise (already a member of a group, or group doesn't exist).

Description

Triggers the group join process. A script should query for group membership to determine when the group has been joined. This function will fail if the user is already a member of a group.

Examples

```
function OnReceivedMessage()
    local msg = ReceivedMessage.GetEventMessage()
    local id = msg:GetType()
    if (id == JOIN_GROUP) then
        groupId = BigInt.Create(64)
        msg:ReadBigInt(groupId)
        joinGroup = true
    end
end

function Update()
    if (joinGroup) then
        joinGroup = false
        if (Group.Join(groupId)) then
            state = JOINING_GROUP
        else
            print("Group join failed!")
            state = JOIN_ERROR      -- could retry, or ask for group ID to be resent
        end
    end
end
```

See Also

- Group.Leave

- Group.GetId
- Group.IsInGroup
- Group.Create

Group.Leave

Brief

Leave a script-created group.

Definition

```
boolean Group.Leave()
```

Arguments

None

Return Values

False if the user is in a non-script created group, or true otherwise.

Description

Removes the local user from a group he or she was previously joined to via script. This function will not remove the user from a group they joined or created themselves.

Examples

```
if (Group.Leave()) then
    state = LEAVING_GROUP
else
    print("User is in their own group!")
end
```

See Also

- Group.IsInGroup
- Group.Create
- Group.Join

GroupDoor

GroupDoor.Create

Brief

Creates a new Group Door.

Definition

```
GroupDoor GroupDoor.Create(string sceneName, boolean isSimpleDoor = true)
```

Arguments

sceneName - the Scene Name of the Group Instance.isSimpleDoor - if set to true (default) this will automate letting the user know if/why they weren't
allowed to enter the door via standard client notifications. Otherwise the user needs to be informed manually,
which will involve using the 'GroupDoor.GetFailedEnterMessage' function. Please see GetFailedEnterMessage
for important details about displaying the message manually.
Note that GroupDoor functionality is currently not available in a personal space. If this function is called
from a personal space, nil is returned.

Return Values

A new Group Door object, or nil on failure.

Description

Creates a new Group Door for accessing a Group Instance, for playing private Group Mini Games.

Examples

```
groupDoor = GroupDoor.Create(sceneName)
groupDoor:Enter()
```

See Also

- GroupDoor.Enter
- GroupDoor.GetFailedEnterMessage

GroupDoor.Enter

Brief

Enters the specified Group Door.

Definition

```
boolean GroupDoor.Enter()
```

Arguments

None

Return Values

True if successful, false otherwise. The requirements for entering a Group Door include being a Member
of a Group, and there being two or more Members in that Group.

Description

This will trigger the relocation system (if the Group Door entry requirements are met). Note: The term 'GroupDoor' is developer-facing terminology and should not be presented to users. When using this feature, please describe it in more user-friendly terms, for example: 'Play this game in a Group Space', 'Play this game with your Group', or 'To play this game with your Group, go through this door'.

Examples

```
groupDoor = GroupDoor.Create(sceneName)
groupDoor:Enter()
```

See Also

- GroupDoor.Create
- GroupDoor.GetFailedEnterMessage

GroupDoor.GetFailedEnterMessage

Brief

Provides a localized string, for use by the developer, to inform the user why an attempt to enter the associated Group Door failed.

Definition

```
string GroupDoor.GetFailedEnterMessage()
```

Arguments

None

Return Values

The error message.

Description

If deciding to manually present this information, this string can be used. The string returned is localized correctly for the current user. It's important to note that it should always be assumed that the message string could be of any length. Messages may vary due to additions, updates, and the language used. The way the message is presented should take this into account, with formatting and scrolling as appropriate.

Examples

```
groupDoor = GroupDoor.Create(sceneName, false)
if (not groupDoor:Enter()) then
    popupMessage = groupDoor:GetFailedEnterMessage()
end
```

See Also

- GroupDoor.Create
- GroupDoor.Enter

Hash

Hash.Create

Brief

Creates a new hash object.

Definition

```
Hash.Create( string type )
```

Arguments

type - The hash type

Return Values

A new instance of a hash object.

Description

Creates a new hash object. Supported types are "SHA1", "MD5".

Examples

```
myHash = Hash.Create( "SHA1" )
```

See Also

Hash.Finalize

Brief

Finalize the hash calculation and return the result.

Definition

```
Hash.Finalize()
```

Arguments

None

Return Values

string - The hash result.

Description

Finalize the hash calculation and return the result.

Examples

```
local result = myHash.Finalize()
```

See Also

Hash.Start

Brief

Prepares a hash object ready for calculating a new hash.

Definition

```
Hash.Start()
```

Arguments

None

Return Values

None

Description

Prepares a hash object ready for calculating a new hash.

Examples

```
myHash.Start()
```

See Also

Hash.Update

Brief

Updates a hash calculation.

Definition

```
Hash.Update( variant data )
```

Arguments

data - A string, number or MemoryContainer

Return Values

None

Description

Updates a hash calculation. Numbers will be hashed as floats.

Examples

```
myHash.Update( "My String" )
```

See Also

HelpSystem

HelpSystem.GetObjectHelpText

Brief

Retrieve the help text associated with an object.

Definition

```
HelpSystem.GetObjectHelpText()
```

Arguments

objectId - The ID of the object to retrieve help text for (string)

Return Values

String if help text exists, otherwise nil

Description

Retrieve the help text associated with

Examples

```
helpText = HelpSystem.GetObjectHelpText( "12345678-12345678-12345678-12345678" )
```

See Also

None

HttpPostData

HttpPostData.AddDataHeader

Brief

Adds a data header to the post data.

Definition

```
boolean HttpPostData.AddDataHeader( HttpPostData postData, string key, string mimeType = nil, string
filename )
```

Arguments

postData - The instance of the post data.key - The name of the value to be added to the post data.mimeType - Optional. The mime type of the data to add to the post data (e.g. "text/xml").filename - Optional. A file name to set for the data. May be needed for the webserver to see the data as a file

Return Values

True if the header was added.

Description

Adds a data header to the post data. This assumes the the script will add the actual data directly in some other way such as writing an XML file into the container. The data must be finalized using FinalizeData before any new values can be added to the post data.

Examples

```
success = HttpPostData.AddDataHeader( postData, "file", "text/xml" )
```

See Also

- HttpPostData.FinalizeData

HttpPostData.AddHttpHeader

Brief

Adds an http header key value pair.

Definition

```
HttpPostData.AddHttpHeader( HttpPostData postData, string key, boolean|number|string|nil value )
```

Arguments

postData - The instance of the post data.key - The name of the value to be added to the header.value - A boolean/number/string/nil to use as the value to add to the header.

Return Values

Description

Adds a value to the http header. The function will error if there is not enough space in the container to add the value. Headers must be added before adding of any data, signify the of the header section by calling HttpPostData.FinalizeHttpHeaders.

Examples

```
container = MemoryContainer.Create( 4096 )

postdata = HttpPostData.Create( container, "multipart/form-data" )

postdata:AddHttpHeader( "key1", "value1" )
postdata:AddHttpHeader( "key2", "value2" )
postdata:AddHttpHeader( "key3", "value3" )
postdata:FinalizeHttpHeaders()

postdata:AddValue( "name", "Bob" )
postdata:AddValue( "password", "abc123" )

postdata:Finalize()
```

See Also

HttpPostData.AddValue

Brief

Adds a value to the post data.

Definition

```
boolean HttpPostData.AddValue( HttpPostData postData, string key, variant value )
```

Arguments

postData - The instance of the post data.key - The name of the value to be added to the post data.value - A boolean/number/string to use as the value to add to the post data.

Return Values

True if the value was added.

Description

Adds a value to the post data. The function will return false if there is not enough space in the container to add the value.

Examples

```
success = HttpPostData.AddValue( postData, "name", "Bob" )
```

See Also

None

HttpPostData.Create

Brief

Creates a new HttpPostData object.

Definition

```
HttpPostData.Create( MemoryContainer container, string contentType )
```

Arguments

container - The memory container to write the post data to.contentType - The encoding used for the post data. Specify either "application/x-www-form-urlencoded" or "multipart/form-data".

Return Values

HttpPostData - A new instance of post data.

Description

Creates a new HttpPostData object. The memory container used will be reset so an data held in it will be lost.

Examples

```
container = MemoryContainer.Create( 4096 )
postData = HttpPostData.Create( container )
```

See Also

HttpPostData.Finalize

Brief

Finalizes the post data.

Definition

```
boolean HttpPostData.Finalize( HttpPostData postData )
```

Arguments

postData - The instance of the post data.

Return Values

True if the post data was finalized.

Description

Finalizes the post data so that it can be sent up to a server. No new data can be added after this call.

Examples

```
success = HttpPostData.Finalize( postData )
```

See Also

- HttpPostData.AddDataHeader

HttpPostData.FinalizeData

Brief

Finalizes any data that has been added after calling AddDataHeader.

Definition

```
boolean HttpPostData.FinalizeData( HttpPostData postData )
```

Arguments

postData - The instance of the post data.

Return Values

True if the data value was finalized.

Description

Finalizes any data that has been added after calling AddDataHeader so that new data values can be added to the post data.

Examples

```
success = HttpPostData.FinalizeData( postData )
```

See Also

- HttpPostData.AddDataHeader

HttpPostData.FinalizeHttpHeaders

Brief

Finalizes the header section after calling AddHttpHeader.

Definition

```
HttpPostData.FinalizeHttpHeaders( HttpPostData postData )
```

Arguments

postData - The instance of the post data.

Return Values

Description

Finalizes the header section after calling AddHttpHeader. No further headers can be added after calling this function.

Examples

```
HttpPostData.FinalizeHttpHeaders( postData )
```

See Also

- HttpPostData.AddHttpHeader

HttpPostData.Reset

Brief

Resets the post data.

Definition

```
HttpPostData.Reset( HttpPostData postData )
```

Arguments

postData - The instance of the post data.

Return Values

None.

Description

Resets the post data so that a new request can be created using the same object.

Examples

```
HttpPostData.Reset( postData )
```

See Also

None

Intersect

Intersect.AabbAabb2d

Brief

Intersect two AABBs in 2D.

Definition

```
boolean [, number, number] Intersect.AabbAabb2d(Vector4 box1min, Vector4 box1max, Vector4 box2min,
Vector4 box2max, boolean pd = false)
boolean [, number, number] Intersect.AabbAabb2d(number b1minX, number b1minY, number b1maxX, number
b1maxY,
                                                number b2minX, number b2minY, number b2maxX, number
b2maxY, boolean pd = false)
```

Arguments

box1Min - Minimum extents of AABB 1.box1Max - Maximum extents of AABB 1.box2Min - Minimum extents of AABB 2.box2Max - Maximum extents of AABB 2.b1minX - Minimum extent on X axis of AABB 1.b1maxX - Maximum extent on X axis of AABB 1.b1minY - Minimum extent on Y axis of AABB 1.b1maxY - Maximum extent on Y axis of AABB 1.b2minX - Minimum extent on X axis of AABB 2.b2maxX - Maximum extent on X axis of AABB 2.b2minY - Minimum extent on Y axis of AABB 2.b2maxY - Maximum extent on Y axis of AABB 2.pd - True to calculate and return penetration distances, or false otherwise.

Return Values

True if the boxes overlap and false if not, and if pd is specified, additionally returns the maximum penetration depth on the x and y axes in that order.

Description

Intersect two axis-aligned bounding boxes in 2D.

Examples

```
box1Min = Vector4.Create(1, 0.5)
box1Max = Vector4.Create(3, 1.5)
box2Min = Vector4.Create(2.5, 1)
box2Max = Vector4.Create(5, 3)
intersect, pdx, pdy = Intersect.AabbAabb2d(box1Min, box1Max, box2Min, box2Max, true)
if (intersect) then
    print("Intersection, penetration axis", pdx <= pdy and "X" or "Y")
end
```

See Also

None

Intersect.AabbAabb3d

Brief

Intersect two AABBs in 3D.

Definition

```
boolean [, number, number, number] Intersect.AabbAabb3d(Vector4 box1Min, Vector4 box1Max, Vector4
box2Min, Vector4 box2Max, boolean pd)
```

Arguments

box1Min - Minimum extents of AABB 1.box1Max - Maximum extents of AABB 1.box2Min - Minimum extents of AABB 2.box2Max - Maximum extents of AABB 2.pd - Calculate and return penetration distance on each axis.

Return Values

True if the boxes intersect, or false otherwise. Additionally, if the parameter pd is true, the penetration distance on each axis (X, Y, Z) is returned, in that order.

Description

Intersect two axis-aligned bounding boxes in 3D.

Examples

```
box1Min = Vector4.Create(1, 0.5, 1)
box1Max = Vector4.Create(3, 1.5, 2)
box2Min = Vector4.Create(2.5, 1, 0)
box2Max = Vector4.Create(5, 3, 1)
intersect, pdx, pdy, pdz = Intersect.AabbAabb3d(box1Min, box1Max, box2Min, box2Max, true)
if (intersect) then
    print("Intersection, max penetration", math.max(pdx, pdy, pdz))
end
```

See Also

None

Intersect.AabbPlane3d

Brief

Intersect an AABB with a plane in 3D.

Definition

```
IntersectSide, number Intersect.AabbPlane3d(Vector4 boxMin, Vector4 boxMax, Vector4 plane)
```

Arguments

boxMin - the minimum extents of the AABB.boxMax - the maximum extents of the AABB.plane - plane equation [normal, distance].

Return Values

The classification of the box relative to the plane and the penetration distance of the AABB relative to the plane:
IntersectSide.Positive - In the positive half-space of the plane.
IntersectSide.Intersect - Overlapping (intersecting) the plane.
IntersectSide.Negative - In the negative half-space of the plane.

Description

The AABB is classified as either fully in the positive halfspace of the plane, fully in the negative halfspace, or overlapping the plane. The penetration distance is also given, which will be positive if any part of the box lies in the negative halfspace of the plane, or negative if fully in the positive halfspace. The penetration distance is given along the plane normal.

Examples

```
boxMin = Vector4.Create(1, 0.5, 0)
boxMax = Vector4.Create(3, 1.5, 0)
if (Intersect.AabbPlane3d(boxMin, boxMax, Vector4.Create(0, -1, 0, 1)) == IntersectSide.Intersect)
then
    print("Intersection!")
end
```

See Also

None

Intersect.CircleAabb2d

Brief

Intersect a circle and an AABB in 2D.

Definition

```
boolean Intersect.CircleAabb2d(Vector4 center, number radius, Vector4 boxMin, Vector4 boxMax)
boolean Intersect.CircleAabb2d(number centerX, number centerY, number radius, number boxMinX, number
boxMinY, number boxMaxX, number boxMaxY)
```

Arguments

center - Circle center point.radius - Circle radius.boxMin - AABB minimum extents.boxMax - AABB maximum extents.centerX - Circle center X coordinate.centerY - Circle center Y coordinate.boxMinX - AABB minimum extent on X axis.boxMinY - AABB minimum extent on Y axis.boxMaxX - AABB maximum extent on X axis.boxMaxY - AABB maximum extent on Y axis.

Return Values

True if the circle and AABB overlap, or false if not.

Description

Intersect a circle and an AABB in 2D.

Examples

```
center = Vector4.Create(0, 1)
radius = 2
boxMin = Vector4.Create(1, 0.5)
boxMax = Vector4.Create(3, 1.5)
if (Intersect.CircleAabb2d(center, radius, boxMin, boxMax)) then
    print("Intersection!")
end
```

See Also

None

Intersect.CircleCircle2d

Brief

Intersect two circles in 2D.

Definition

```
boolean [, number] Intersect.CircleCircle2d(Vector4 center1, number radius1, Vector4 center2, number
radius2, boolean pd = false)
```

Arguments

center1 - center point of circle 1.radius1 - radius of circle 1.center2 - center point of circle 2.radius2 - radius of circle 2.pd - return penetration distance.

Return Values

True if circles intersect or false otherwise. If pd is true, additionally returns the penetration distance of the circles.

Description

Intersect two circles in 2D. If pd is true, also returns the penetration distance, which is >= 0 if the circles intersect, or negative otherwise. If the penetration distance is greater than the smaller radius of the two, the small circle lies wholly inside the large one. The penetration distance lies along the vector between the circles' centers.

Examples

```
local intersect, pd = Intersect.CircleCircle2d(center1, radius1, center2, radius2, true)
if (intersect) then
    if (pd >= math.min(radius1, radius2)) then
        print("Big circle wholly contains smaller circle")
    else
        print("Circles intersect with penetration distance", pd)
    end
end
```

See Also

None

Intersect.DistSqLineLine3d

Brief

Calculate the squared distance between two lines in 3D at their point of closest approach.

Definition

```
number [, number, number] Intersect.DistSqLineLine3d(Vector4 line1Point, Vector4 line1Dir, Vector4
line2Point, Vector4 line2Dir, boolean closest = false)
```

Arguments

line1Point - a point on line 1.line1Dir - the normalized direction of line 1.line2Point - a point on line 2.line2Dir - the normalized direction of line 2.closest - If true, return the distance parameters for each line at the point of closest approach.

Return Values

The squared distance between the two lines at their point of closest approach.

Description

Two lines in 3D will rarely intersect exactly, so it will probably be more valuable to know the distance between the lines at their closest point. This function will return the square of that distance, which can be compared to a value suitable to the application. Note that squared distances can quickly exceed the limits of a floating point number, so it may be best to translate the lines to the origin before calling this function.

Examples

```
line1Point = Vector4.Create(0, 1, 0)
line1Dir = Vector4.Create(1, 0, 0)
line2Point = Vector4.Create(0, 2, 0)
line2Dir = Vector4.Create(1, 0, 0)
dist, t1, t2 = Intersect.DistSqLineLine3d(line1Point, line1Dir, line2Point, line2Dir, true)
print("Closest approach distance is", dist, "and closet point of approach is", line1Point +
t1*line1Dir)
```

See Also

None

Intersect.LineAabb2d

Brief

Intersect a line with an AABB in 2D.

Definition

```
boolean, number, number Intersect.LineAabb2d(Vector4 linePoint, Vector4 lineDir, Vector4 boxMin,
Vector4 boxMax)
boolean, number, number Intersect.LineAabb2d(number linePointX, number linePointY, number lineDirX,
number lineDirY,
                            number boxMinX, number boxMinY, number boxMaxX, number boxMaxY)
```

Arguments

linePoint - a point on the line.lineDir - normalized line direction.boxMin - minimum extent of AABB.boxMax - maximum extent of AABB.linePointX - line point X coordinate.linePointY - line point Y coordinate.lineDirX - normalized line direction X coordinate.lineDirY - normalized line direction Y coordinate.boxMinX - minimum X extent of AABB.boxMinY - minimum Y extent of AABB.boxMaxX - maximum X extent of AABB.boxMaxY - maximum Y extent of AABB.

Return Values

True if the line intersects or false otherwise, and if true, the intersection distances on the line.

Description

Intersect a line with an AABB in 2D. If the line intersects the AABB, the intersection parameters are returned also. To compute an intersection point from these parameters, compute linePoint + t*lineDir, where t is the intersection distance returned from the function.

Examples

```
linePoint = Vector4.Create(0, 1)
lineDir = Vector4.Create(1, 0)
boxMin = Vector4.Create(1, 0.5)
boxMax = Vector4.Create(3, 1.5)
intersect, t1, t2 = Intersect.LineAabb2d(linePoint, lineDir, boxMin, boxMax)
if (intersect) then
    print("Intersection 1 at", t1)
    -- t2 can be same as t1 if intersects a corner
    if (t2 ~= t1) then
        print("Intersection 2 at", t2)
    end
end
```

See Also

- Intersect.RayAabb2d
- Intersect.SegmentAabb2d

Intersect.LineAabb3d

Brief

Intersect a line with an AABB in 3D.

Definition

```
boolean, number, number Intersect.LineAabb3d(Vector4 linePoint, Vector4 lineDir, Vector4 boxMin,
Vector4 boxMax)
```

Arguments

linePoint - a point on the line.lineDir - normalized line direction.boxMin - minimum extent of AABB.boxMax - maximum extent of AABB.

Return Values

True if the line intersects or false otherwise, and if true, the intersection distances on the line.

Description

Intersect a line with an AABB in 3D. If the line intersects the AABB, the intersection parameters are returned also. To compute an intersection point from these parameters, compute linePoint + t*lineDir, where t is the intersection distance returned from the function.

Examples

```
linePoint = Vector4.Create(0, 1, 0)
lineDir = Vector4.Create(1, 0, 0)
boxMin = Vector4.Create(1, 0.5, 0)
boxMax = Vector4.Create(3, 1.5, 0)
intersect, t1, t2 = Intersect.LineAabb3d(linePoint, lineDir, boxMin, boxMax)
if (intersect) then
    print("Intersection 1 at", t1)
    -- t2 can be same as t1 if intersects a corner
    if (t2 ~= t1) then
        print("Intersection 2 at", t2)
    end
end
```

See Also

- Intersect.RayAabb3d
- Intersect.SegmentAabb3d

Intersect.LineCircle2d

Brief

Intersect a line with a circle in 2D.

Definition

```
boolean [, number [, number]] Intersect.LineCircle2d(Vector4 linePoint, Vector4 lineDir, Vector4
center, number radius, boolean ip = false)
boolean [, number [, number]] Intersect.LineCircle2d(number linePointX, number linePointY, number
lineDirX, number lineDirY,
                                                     number centerX, number centerY, number radius,
boolean ip = false)
```

Arguments

linePoint - a point on the line (any point).lineDir - normalized direction vector of line (sign is unimportant).center - center point of circle.radius - circle radius.linePointX - X coordinate of a point on the line.linePointY - Y coordinate of a point on the line.lineDirX - X coordinate of normalized direction vector of line.lineDirY - Y coordinate of normalized direction vector of line.centerX - circle center X coordinate.centerY - circle center Y coordinate.ip - true to return intersection distances, false if not required.

Return Values

True if the line intersects the circle, or false otherwise. If ip is specified, additionally returns either 0, 1 or 2 intersection distances, for the point(s) at which the line intersects the circle, if any.

Description

Intersect a line with a circle in 2D, and if requested return the distances along the line of the intersection point(s). If the intersection distance is not requested it will not be computed, which will be more efficient. Note that a line may intersect a circle at 0 (no intersection), 1 (tangential intersection) or 2 points (all other cases). The number of return values will reflect these cases. Note that a line is considered infinite and thus the "order" of intersection points is not considered. If you wish to interpret the intersection distances in terms of distance then you will need to perform this yourself.

Examples

```
linePoint = Vector4.Create(0, 1)
lineDir = Vector4.Create(1, 0)
center = Vector4.Create(3, 1)
intersect, t1, t2 = Intersect.LineCircle2d(linePoint, lineDir, center, 2, true)
if (intersect) then
    -- t1 must be valid
    print("Intersection 1 at", t1)
    -- t2 may be equal to t1 if line intersects tangentially
    if (t2 ~= t1) then
        print("Intersection 2 at", t2)
    end
end
```

See Also

- Intersect.RayCircle2d
- Intersect.SegmentCircle2d

Intersect.LineLine2d

Brief

Intersect two lines in 2D.

Definition

```
boolean [, number, number] Intersect.LineLine2d(Vector4 line1Point, Vector4 line1Dir, Vector4
line2Point, Vector4 line2Dir, boolean ip = false)
boolean [, number, number] Intersect.LineLine2d(number line1px, number line1py, number line1dx, number
line1dy,
                                                number line2px, number line2py, number line2dx, number
line2dy, boolean ip = false)
```

Arguments

line1Point - a point on line 1 (any point).line1Dir - normalized direction vector of line 1 (sign is unimportant).line2Point - a point on line 2 (any point).line2Dir - normalized direction vector of line 2 (sign is unimportant).line1PointX - X coordinate of a point on line 1 (any point).line1PointY - Y coordinate of a point on line 1 (any point).line1DirX - X coordinate of normalized direction vector of line 1 (sign is unimportant).line1DirY - Y coordinate of normalized direction vector of line 1 (sign is unimportant).line2PointX - X coordinate of a point on line 2 (any point).line2PointY - Y coordinate of a point on line 2 (any point).line2DirX - X coordinate of normalized direction vector of line 2 (sign is unimportant).line2DirY - Y coordinate of normalized direction vector of line 2 (sign is unimportant).ip - true to return intersection distances, false if not required.

Return Values

True if the lines intersect, false otherwise. If ip is specified, additionally returns the intersection distance for both lines.

Description

Intersect two lines in 2D, and if requested return the intersection distance for both lines if they intersect. Requesting the intersection distances has no additional computation cost for this function. If the intersection point coordinates are required, it can be computed with linePoint + t*lineDir, where t is the intersection distance for that line.

Examples

```
line1Point = Vector4.Create(1, 1)
line1Dir = Vector4.Create(1, 0)
line2Point = Vector4.Create(3, 0)
line2Dir = Vector4.Create(0, 1)
intersect, t1, t2 = Intersect.LineLine2d(line1Point, line1Dir, line2Point, line2Dir, true)
if (intersect) then
    ip = line1Point + t1 * line1Dir     -- or line2Point + t2 * line2Dir
end
```

See Also

- Intersect.RayRay2d
- Intersect.RayLine2d
- Intersect.LineSegment2d
- Intersect.RaySegment2d
- Intersect.SegmentSegment2d

Intersect.LinePlane3d

Brief

Intersect a line with a plane in 3D.

Definition

```
IntersectType [ , number ] Intersect.LinePlane3d(Vector4 linePoint, Vector4 lineDir, Vector4 plane)
```

Arguments

linePoint - a point on the line.lineDir - direction vector of the line.plane - plane equation [normal, distance]

Return Values

If the line intersects the plane, IntersectType.Intersect is returned, if not, IntersectType.Disjoint is returned.
If the line lies in the plane, IntersectType.Overlap is returned. If the result is IntersectType.Intersect, the intersection parameter t is returned (intersection point = linePoint + t*lineDir). Other results have no additional return values.

Description

A line is directionless, so all queries are effectively "two-sided". If Intersect.Disjoint is returned, this indicates the line is parallel to the plane (perpendicular to the plane normal).

Examples

```
if (Intersect.LinePlane3d(linePoint, lineDir, plane) ~= IntersectType.Disjoint) then
    print("Line intersects plane")
end
```

See Also

- Intersect.RayPlane3d

- Intersect.SegmentPlane3d

Intersect.LineSegment2d

Brief

Intersect a line and a segment in 2D.

Definition

```
boolean [, number, number] Intersect.LineSegment2d(Vector4 linePoint, Vector4 lineDir, Vector4
segStart, Vector4 segVec, boolean ip = false)
boolean [, number, number] Intersect.LineSegment2d(number linePointX, number linePointY, number
lineDirX, number lineDirY,
                                                   number segStartX, number segStartY, number segVecX,
number segVecY, boolean ip = false)
```

Arguments

linePoint - a point on the line (any point).lineDir - normalized direction vector of line (sign is unimportant).segStart - first (start) point of segment.segVec - vector from first (start) point of segment to second (end) point of segment. If the segment points are A and B, the vector is B-A.linePointX - X coordinate of a point on the line.linePointY - Y coordinate of a point on the line.lineDirX - X coordinate of normalized direction vector of line.lineDirY - Y coordinate of normalized direction vector of line.segStartX - X coordinate of segment start point.segStartY - Y coordinate of segment start point.segVecX - X coordinate of segment direction vector (see above).segVecY - Y coordinate of segment direction vector (see above).ip - true to return intersection distances, false if not required.

Return Values

True if the line and segment intersect, false otherwise. If ip is specified, additionally returns the intersection distance for the line and segment, in that order.

Description

Intersect a line and a segment in 2D, and if requested return the intersection distance for both line and segment if they intersect. Requesting the intersection distances has no additional computation cost for this function. If the intersection point coordinates are required, it can be computed with linePoint + t*lineDir, where t is the intersection distance for the line, or segStart + u*segVec, where u is the intersection distance for the segment.

Examples

```
linePoint = Vector4.Create(1, 1)
lineDir = Vector4.Create(1, 0)
segmentStart = Vector4.Create(3, 0)
segmentVec = Vector4.Create(0, 5)
intersect, t1, t2 = Intersect.LineSegment2d(linePoint, lineDir, segmentStart, segmentVec, true)
if (intersect) then
    ip = linePoint + t1 * lineDir       -- or segmentStart + t2 * segmentVec
end
```

See Also

- Intersect.RayRay2d
- Intersect.RayLine2d
- Intersect.RaySegment2d
- Intersect.LineLine2d
- Intersect.SegmentSegment2d

Intersect.LineSphere3d

Brief

Intersect a line with a sphere in 3D.

Definition

```
boolean [, number, number] Intersect.LineSphere3d(Vector4 linePoint, Vector4 lineDir, Vector4 center,
number radius, boolean ip = false)
```

Arguments

linePoint - any point on the line.lineDir - line direction (assumed to be normalized).center - center point of sphere.radius - radius of sphere.ip - if true, compute and return intersection points.

Return Values

True if the line intersects the sphere, false otherwise. If ip is specified, the two intersection distances are returned additionally.

Description

Intersect a line with a sphere in 3D, and if requested return the intersection distances along the line of the intersection points. If the intersection distance is not requested it will not be computed, which will be more efficient. If the intersection points in space are desired, then the points would be computed as linePoint + t*lineDir, with t the intersection distance. Note that a line can either not intersect, intersect at two points, or intersect at one point. In the latter case, both intersection distances will be the same. The ordering of the returned intersection values is not defined as a line is not directed, unlike a ray, and the sphere is considered hollow for the purposes of intersection.

Examples

```
linePoint = Vector4.Create(0, 1, 0)
lineDir = Vector4.Create(1, 0, 0)
center = Vector4.Create(3, 1, 0)
intersect, t1, t2 = Intersect.LineSphere3d(linePoint, lineDir, center, 2, true)
if (intersect) then
    -- t1 must be valid
    print("Intersection 1 at", t1)
    -- t2 be equal to t1 if line intersects tangentially
    if (t2 ~= t1) then
        print("Intersection 2 at", t2)
    end
end
```

See Also

- Intersect.SegmentSphere3d

Intersect.PlanePlane3d

Brief

Intersect two planes in 3D.

Definition

```
IntersectType Intersect.PlanePlane3d(Vector4 plane1, Vector4 plane2, Vector4 linePoint = nil, Vector4
lineDir = nil)
```

Arguments

plane1 - first plane [normal, distance].plane2 - second plane [normal, distance].linePoint - if specified, will be set to a point on the line of intersection of the planes if they intersect.lineDir - if specified, will be set to the normalized direction of the line of intersection of the planes if they intersect.

Return Values

IntersectType.Intersect if the planes are non-parallel. If co-planar, IntersectType.Overlap is returned, or IntersectType.Disjoint otherwise.

Description

Intersects two planes in 3D. Unless parallel, two planes will always intersect at a line. If the line of intersection is required (if it exists), pass Vector4 instances to linePoint and/or lineDir. Not requesting the line of intersection will be more efficient.

Examples

```
linePoint = Vector4.Create()
lineDir = Vector4.Create()
if (Intersect.PlanePlane3d(Vector4.Create(0, 1, 0, 3), Vector4.Create(1, 0, 0, 5), linePoint, lineDir)
== IntersectType.Intersect) then
    print("Lines intersect at line", linePoint, lineDir)
end
```

See Also

None

Intersect.RayAabb2d

Brief

Intersect a ray with an AABB in 2D.

Definition

```
boolean, number, number Intersect.RayAabb2d(Vector4 rayOrig, Vector4 rayDir, Vector4 boxMin, Vector4
boxMax)
boolean, number, number Intersect.RayAabb2d(number rayOrigX, number rayOrigY, number rayDirX, number
rayDirY,
                                            number boxMinX, number boxMinY, number boxMaxX, number
boxMaxY)
```

Arguments

rayOrig - ray origin.rayDir - normalized ray direction.boxMin - minimum extents of AABB.boxMax - maximum extents of AABB.rayOrigX - ray origin X coordinate.rayOrigY - ray origin Y coordinate.rayDirX - normalized ray direction X coordinate.rayDirY - normalized ray direction Y coordinate.boxMinX - minimum X extent of AABB.boxMinY - minimum Y extent of AABB.boxMaxX - maximum X extent of AABB.boxMaxY - maximum Y extent of AABB.

Return Values

True if the ray intersects or false otherwise, and if true, the intersection distance(s) along the ray.

Description

Intersect a ray with an axis-aligned bounding box (AABB) in 2D and return the intersection distance(s) if it intersects. The intersection distances will be returned in order of distance from the ray origin.

Examples

```
rayOrig = Vector4.Create(0, 1)
rayDir = Vector4.Create(1, 0)
boxMin = Vector4.Create(1, 0.5)
boxMax = Vector4.Create(3, 1.5)
intersect, t1, t2 = Intersect.RayAabb2d(rayOrig, rayDir, boxMin, boxMax)
if (intersect) then
    print("First intersection at", t1)
    -- t2 can be nil if ray starts inside box, or same as t1 if intersects a corner
    if (t2 ~= nil and t2 ~= t1) then
        print("Second intersection at", t2)
    end
end
```

See Also

- Intersect.LineAabb2d
- Intersect.SegmentAabb2d

Intersect.RayAabb3d

Brief

Intersect a ray (half-line) with an AABB in 3D.

Definition

```
boolean, number, number Intersect.RayAabb3d(Vector4 rayOrig, Vector4 rayDir, Vector4 boxMin, Vector4
boxMax)
```

Arguments

rayOrig - ray origin.rayDir - normalized ray direction.boxMin - minimum extents of AABB.boxMax - maximum extents of AABB.

Return Values

True if the ray intersects or false otherwise, and if true, the intersection distance(s) along the ray.

Description

Intersect a ray with an AABB in 3D and return the intersection distance(s) if it intersects.

Examples

```
rayOrig = Vector4.Create(0, 1, 0)
rayDir = Vector4.Create(1, 0, 0)
boxMin = Vector4.Create(1, 0.5, 0)
boxMax = Vector4.Create(3, 1.5, 0)
intersect, t1, t2 = Intersect.RayAabb3d(rayOrig, rayDir, boxMin, boxMax)
if (intersect) then
    print("First intersection at", t1)
    -- t2 can be nil if ray starts inside box, or same as t1 if intersects a corner
    if (t2 ~= nil and t2 ~= t1) then
        print("Second intersection at", t2)
    end
end
```

See Also

- Intersect.LineAabb3d

- Intersect.SegmentAabb3d

Intersect.RayCircle2d

Brief

Intersect a ray with a circle in 2D.

Definition

```
boolean [, number [, number]] Intersect.RayCircle2d(Vector4 rayOrig, Vector4 rayDir, Vector4 center,
number radius, boolean ip = false)
boolean [, number [, number]] Intersect.RayCircle2d(number rayOrigX, number rayOrigY, number rayDirX,
number rayDirY,
                                                    number centerX, number centerY, number radius,
boolean ip = false)
```

Arguments

rayOrig - ray origin.rayDir - normalized direction vector of ray.center - center point of circle.radius - circle radius.rayOrigX - ray origin X coordinate.rayOrigY - ray origin Y coordinate.rayDirX - X coordinate of normalized direction vector of ray.rayDirY - Y coordinate of normalized direction vector of ray.centerX - circle center X coordinate.centerY - circle center Y coordinate.ip - true to return intersection distances, false if not required.

Return Values

True if the ray intersects the circle, or false otherwise. If ip is specified, additionally returns either 0, 1 or 2 intersection distances, for the point(s) at which the ray intersects the circle, if any.

Description

Intersect a ray (half-line) with a circle in 2D, and if requested return the distances along the ray of the intersection point(s). If the intersection distance is not requested it will not be computed, which will be more efficient. Note that a ray may intersect a circle at 0 (no intersection), 1 (tangential intersection, or ray starting inside circle) or 2 points (all other cases). The number of return values will reflect these cases.

Examples

```
rayOrigin = Vector4.Create(0, 1)
rayDir = Vector4.Create(1, 0)
center = Vector4.Create(3, 1)
intersect, t1, t2 = Intersect.RayCircle2d(rayOrigin, rayDir, center, 2, true)
if (intersect) then
    -- t1 must be valid
    print("First intersection at", t1)
    -- t2 may be nil (ray starts inside circle), may also be equal to t1 if ray intersects
tangentially
    if (t2 ~= nil and t2 ~= t1) then
        print("Second intersection at", t2)
    end
end
```

See Also

- Intersect.LineCircle2d
- Intersect.SegmentCircle2d

Intersect.RayLine2d

Brief

Intersect a ray (half-line) and a line in 2D.

Definition

```
boolean [, number, number] Intersect.RayLine2d(Vector4 rayOrig, Vector4 rayDir, Vector4 linePoint,
Vector4 lineDir, boolean ip = false)
boolean [, number, number] Intersect.RayLine2d(number rayOrigX, number rayOrigY, number rayDirX,
number rayDirY,
                                               number linePointX, number linePointY, number lineDirX,
number lineDirY, boolean ip = false)
```

Arguments

rayOrig - ray origin.rayDir - normalized direction vector of ray.linePoint - a point on the line (any point).lineDir - normalized direction vector of line (sign is unimportant).rayOrigX - ray origin X coordinate.rayOrigY - ray origin Y coordinate.rayDirX - X coordinate of normalized direction vector of ray.rayDirY - Y coordinate of normalized direction vector of ray.linePointX - X coordinate of a point on the line.linePointY - Y coordinate of a point on the line.lineDirX - X coordinate of normalized direction vector of line.lineDirY - Y coordinate of normalized direction vector of line.ip - true to return intersection distances, false if not required.

Return Values

True if the ray and line intersect, false otherwise. If ip is specified, additionally returns the intersection distance for both the ray and line, in that order.

Description

Intersect a ray (half-line) and a line in 2D, and if requested return the intersection distance for both ray and line if they intersect. Requesting the intersection distances has no additional computation cost for this function. If the intersection point coordinates are required, it can be computed with either rayOrigin + t*rayDir, where t is the intersection distance for the ray, or linePoint + u*lineDir, where u is the intersection distance for the line.

Examples

```
rayOrigin = Vector4.Create(1, 1)
rayDir = Vector4.Create(1, 0)
linePoint = Vector4.Create(3, 0)
lineDir = Vector4.Create(0, 1)
intersect, t1, t2 = Intersect.RayLine2d(rayOrigin, rayDir, linePoint, lineDir, true)
if (intersect) then
    ip = rayOrigin + t1 * rayDir        -- or linePoint + t2 * lineDir
end
```

See Also

- Intersect.RaySegment2d
- Intersect.RayRay2d
- Intersect.LineSegment2d
- Intersect.LineLine2d
- Intersect.SegmentSegment2d

Intersect.RayPlane3d

Brief

Intersect a ray (half-line) with a plane in 3D.

Definition

```
IntersectType [ , number ] Intersect.RayPlane3d(Vector4 rayOrig, Vector4 rayDir, Vector4 plane, bool
twoSided = false)
```

Arguments

rayOrig - ray origin.rayDir - ray direction (assumed to be normalized).plane - plane equation [normal, distance]twoSided - if true, will check for intersection with the "back" side of the plane (ray origin in negative halfspace)

Return Values

If the ray intersects the plane, IntersectType.Intersect is returned, if not, IntersectType.Disjoint is returned.
If the ray lies in the plane, IntersectType.Overlap is returned. If the result is IntersectType.Intersect, the intersection distance along the ray is returned additionally. Other results have no additional return values.

Description

Intersect a ray (half-line) with a plane in 3D. The plane can be considered as either 1- or 2-sided using the twoSided parameter: 1-sided planes will not consider intersections from rays originating in the negative halfspace.

Examples

```
if (Intersect.RayPlane3d(rayOrig, rayDir, plane) ~= IntersectType.Disjoint) then
    print("Ray intersects plane")
end
```

See Also

- Intersect.LinePlane3d
- Intersect.SegmentPlane3d

Intersect.RayRay2d

Brief

Intersect two rays (half-lines) in 2D.

Definition

```
boolean [, number, number] Intersect.RayRay2d(Vector4 ray1Orig, Vector4 ray1Dir, Vector4 ray2Orig,
Vector4 ray2Dir, boolean ip = false)
boolean [, number, number] Intersect.RayRay2d(number ray1OrigX, number ray1OrigY, number ray1DirX,
number ray1DirY,
                                              number ray2OrigX, number ray2OrigY, number ray2DirX,
number ray2DirY, boolean ip = false)
```

Arguments

ray1Orig - ray 1 origin.ray1Dir - normalized direction vector of ray 1.ray2Orig - ray 2 origin.ray2Dir - normalized direction vector of ray 2.ray1OrigX - ray 1 origin X coordinate.ray1OrigY - ray 1 origin Y coordinate.ray1DirX - X coordinate of normalized direction vector of ray 1.ray1DirY - Y coordinate of normalized direction vector of ray 1.ray2OrigX - ray 2 origin X coordinate.ray2OrigY - ray 2 origin Y coordinate.ray2DirX - X coordinate of normalized direction vector of ray 2.ray2DirY - Y coordinate of normalized direction vector of ray 2.ip - true to return intersection distances, false if not required.

Return Values

True if the rays intersect or false otherwise. If ip is true, additionally returns the distance along the two rays to the intersection point.

Description

Intersect two rays (half-lines) in 2D, and if requested return the intersection distance for both rays if they intersect. Requesting the intersection distances has no additional computation cost for this function. If the intersection point coordinates are required, it can be computed with rayOrigin + t*rayDir, where t is the intersection distance for that ray.

Examples

```
ray1Origin = Vector4.Create(1, 1)
ray1Dir = Vector4.Create(1, 0)
ray2Origin = Vector4.Create(3, 0)
ray2Dir = Vector4.Create(0, 1)
intersect, t1, t2 = Intersect.RayRay2d(ray1Origin, ray1Dir, ray2Origin, ray2Dir, true)
if (intersect) then
    ip = ray1Origin + t1 * ray1Dir      -- or ray2Origin + t2 * ray2Dir
end
```

See Also

- Intersect.RaySegment2d
- Intersect.RayLine2d
- Intersect.LineSegment2d
- Intersect.LineLine2d
- Intersect.SegmentSegment2d

Intersect.RaySegment2d

Brief

Intersect a ray (half-line) and a segment in 2D.

Definition

```
boolean [, number, number] Intersect.RaySegment2d(Vector4 rayOrig, Vector4 rayDir, Vector4 segStart,
Vector4 segVec, boolean ip = false)
boolean [, number, number] Intersect.RaySegment2d(number rayOrigX, number rayOrigY, number rayDirX,
number rayDirY,
                                                  number segStartX, number segStartY, number segVecX,
number segVecY, boolean ip = false)
```

Arguments

rayOrig - ray origin.rayDir - normalized direction vector of ray.segStart - first (start) point of segment.segVec - vector from first (start) point of segment to second (end) point of segment. If the segment points
are A and B, the vector is B-A.rayOrigX - ray origin X coordinate.rayOrigY - ray origin Y coordinate.rayDirX - X coordinate of normalized direction vector of ray.rayDirY - Y coordinate of normalized direction vector of ray.segStartX - X coordinate of segment start point.segStartY - Y coordinate of segment start point.segVecX - X coordinate of segment direction vector (see above).segVecY - Y coordinate of segment direction vector (see above).ip - true to return intersection distances, false if not required.

Return Values

True if the ray and segment intersect, false otherwise. If ip is specified, additionally returns the intersection distance
for both the ray and segment, in that order.

Description

Intersect a ray (half-line) and a segment in 2D, and if requested return the intersection distance for both ray and segment if they intersect. Requesting the intersection distances has no additional computation cost for this function. If the intersection point coordinates are required, it can be computed with either rayOrigin + t*rayDir, where t is the intersection distance for the ray, or segStart + u*segVec, where u is the intersection distance for the segment.

Examples

```
rayOrigin = Vector4.Create(1, 1)
rayDir = Vector4.Create(1, 0)
segmentStart = Vector4.Create(3, 0)
segmentVec = Vector4.Create(0, 5)
intersect, t1, t2 = Intersect.RaySegment2d(rayOrigin, rayDir, segmentStart, segmentVec, true)
if (intersect) then
    ip = rayOrigin + t1 * rayDir        -- or segmentStart + t2 * segmentVec
end
```

See Also

- Intersect.RayRay2d
- Intersect.RayLine2d
- Intersect.LineSegment2d
- Intersect.LineLine2d
- Intersect.SegmentSegment2d

Intersect.RaySphere3d

Brief

Intersect a ray with a sphere in 3D.

Definition

```
boolean [, number, number] Intersect.RaySphere3d(Vector4 rayOrig, Vector4 rayDir, Vector4 center,
number radius, boolean ip = false)
```

Arguments

rayOrig - ray origin.rayDir - ray direction (assumed to be normalized).center - center point of sphere.radius - radius of sphere.ip - if true, compute and return intersection distances.

Return Values

True if the ray intersects the sphere, false otherwise. If ip is specified, the intersection distances
along the ray are returned. The first return value is the first intersection, and the second the final.

Description

Intersect a ray (half-line) with a sphere in 3D, and if requested return the distances along the ray of the intersection points. If "ip" is true the intersection distances will be returned after the boolean return value. If the intersection distance is not requested it will not be computed, which will be more efficient. If the intersection points in space are desired, they can be computed with rayOrig + t*rayDir, where t is the intersection distance. Note that a ray can either not intersect, intersect at two points, or intersect at one point. In the latter case, both intersection distances will be the same. The sphere is considered "solid", so a ray originating inside the sphere will have an initial intersection distance of 0.

Examples

```
rayOrigin = Vector4.Create(0, 1, 0)
rayDir = Vector4.Create(1, 0, 0)
center = Vector4.Create(3, 1, 0)
intersect, t1, t2 = Intersect.RaySphere3d(rayOrigin, rayDir, center, 2, true)
if (intersect) then
    -- t1 must be valid
    print("First intersection at", t1)
    -- t2 may be nil (ray starts inside sphere), may also be equal to t1 if ray intersects
tangentially
    if (t2 ~= nil and t2 ~= t1) then
        print("Second intersection at", t2)
    end
end
```

See Also

- Intersect.LineSphere3d
- Intersect.SegmentSphere3d

Intersect.SegmentAabb2d

Brief

Intersect a segment with an AABB in 2D.

Definition

```
boolean [, number [, number]] Intersect.SegmentAabb2d(Vector4 segStart, Vector4 segVec, Vector4
boxMin, Vector4 boxMax)
boolean [, number [, number]] Intersect.SegmentAabb2d(number segStartX, number segStartY, number
segVecX, number segVecY,
                                                      number boxMinX, number boxMinY, number boxMaxX,
number boxMaxY)
```

Arguments

segStart - segment start point.segVec - vector from segment start point to end point.boxMin - minimum extents of AABB.boxMax - maximum extents of AABB.segStartX - segment start point X coordinate.segStartY - segment start point Y coordinate.segVecX - segment vector X coordinate.segVecY - segment vector Y coordinate.boxMinX - minimum X extent of AABB.boxMinY - minimum Y extent of AABB.boxMaxX - maximum X extent of AABB.boxMaxY - maximum Y extent of AABB.

Return Values

True if the segment intersects or false otherwise, and if true, the intersection distance(s) along the segment.

Description

Intersect a segment with an AABB in 2D and return the intersection distance(s) if it intersects. To compute an intersection point from these parameters, compute segStart + t*segVec, where t is an intersection distance returned from the function.

Examples

```
segStart = Vector4.Create(0, 1)
segVec = Vector4.Create(6, 0)
boxMin = Vector4.Create(1, 0.5)
boxMax = Vector4.Create(3, 1.5)
intersect, t1, t2 = Intersect.SegmentAabb2d(segStart, segVec, boxMin, boxMax)
if (intersect) then
    print("First intersection at", t1)
    -- t2 can be nil (if segment starts inside box or doesn't cross box), or same as t1 if intersects
a corner
    if (t2 ~= nil and t2 ~= t1) then
        print("Second intersection at", t2)
    end
end
```

See Also

- Intersect.LineAabb2d
- Intersect.RayAabb2d

Intersect.SegmentAabb3d

Brief

Intersect a segment with an AABB in 3D.

Definition

```
boolean [, number [, number]] Intersect.SegmentAabb3d(Vector4 segStart, Vector4 segVec, Vector4
boxMin, Vector4 boxMax)
```

Arguments

segStart - segment start point.segVec - vector from segment start point to end point.boxMin - minimum extents of AABB.boxMax - maximum extents of AABB.

Return Values

True if the segment intersects or false otherwise, and if true, the intersection distance(s) along the segment.

Description

Intersect a segment with an AABB in 3D and return the intersection distance(s) if it intersects. To compute an intersection point from these parameters, compute segStart + t*segVec, where t is an intersection distance returned from the function.

Examples

```
segStart = Vector4.Create(0, 1, 0)
segVec = Vector4.Create(6, 0, 0)
boxMin = Vector4.Create(1, 0.5, 0)
boxMax = Vector4.Create(3, 1.5, 0)
intersect, t1, t2 = Intersect.SegmentAabb2d(segStart, segVec, boxMin, boxMax)
if (intersect) then
    print("First intersection at", t1)
    -- t2 can be nil (if segment starts inside box or doesn't cross box), or same as t1 if intersects
a corner
    if (t2 ~= nil and t2 ~= t1) then
        print("Second intersection at", t2)
    end
end
```

See Also

- Intersect.RayAabb3d
- Intersect.LineAabb3d

Intersect.SegmentCircle2d

Brief

Intersect a segment with a circle in 2D.

Definition

```
boolean [, number [, number]] Intersect.SegmentCircle2d(Vector4 segStart, Vector4 segVec, Vector4
center, number radius, boolean ip = false)
boolean [, number [, number]] Intersect.SegmentCircle2d(number segStartX, number segStartY, number
segVecX, number segVecY, number centerX, number centerY, number radius, boolean ip = false)
```

Arguments

segStart - first (start) point of segment.segVec - vector from first (start) point of segment to second (end) point of segment. If the segment points are A and B, the vector is B-A.center - center point of circle.radius - radius of circle.segStartX - X coordinate of segment start.segStartY - Y coordinate of segment start.segVecX - X coordinate of segment vector (see segVec).segVecY - Y coordinate of segment vector (see segVec).centerX - X coordinate of circle center.centerY - Y coordinate of circle center.ip - true to return intersection distances, false if not required.

Return Values

True if the segment intersects the circle, or false otherwise. If ip is specified, additionally returns either 0, 1 or 2 intersection distances, for the point(s) at which the segment intersects the circle, if any.

Description

Intersect a segment with a circle in 2D, and if requested return the distances along the segment of the intersection point(s). If the intersection distance is not requested it will not be computed, which will be more efficient. Note that a segment may intersect a circle at 0 (no intersection, including segment wholly inside the circle), 1 (tangential intersection, or segment straddles circle boundary) or 2 points (all other cases). The number of return values will reflect these cases.

Examples

```
segStart = Vector4.Create(0, 1)
segVec = Vector4.Create(6, 0)
center = Vector4.Create(3, 1)
intersect, t1, t2 = Intersect.SegmentCircle2d(segStart, segVec, center, 2, true)
if (intersect) then
    -- t1 must be valid
    print("First intersection at", t1)
    -- t2 may be nil (segment starts inside circle or doesn't cross circle), may also be equal to t1
if segment intersects tangentially
    if (t2 ~= nil and t2 ~= t1) then
        print("Second intersection at", t2)
    end
end
```

See Also

- Intersect.RayCircle2d
- Intersect.LineCircle2d

Intersect.SegmentPlane3d

Brief

Intersect a segment with a plane in 3D.

Definition

```
IntersectType [ , number ] Intersect.SegmentPlane3d(Vector4 segStart, Vector4 segVec, Vector4 plane,
bool twoSided = false)
```

Arguments

segStart - first segment point.segVec - vector from first segment point to second segment point.plane - plane equation [normal, distance]twoSided - if true, will check for intersection with the "back" side of the plane (segment start in negative halfspace)

Return Values

If the segment intersects the plane, IntersectType.Intersect is returned, if not, IntersectType.Disjoint is returned.
If the segment lies in the plane, IntersectType.Overlap is returned. If the result is IntersectType.Intersect, the intersection distance along the segment is returned additionally. Other results have no additional return values.

Description

Intersect a line segment with a plane in 3D. The plane can be considered as either 1- or 2-sided using the twoSided parameter: 1-sided planes will not consider intersections from segments originating in the negative halfspace.

Examples

```
if (Intersect.SegmentPlane3d(segStart, segEnd - segStart, plane) ~= IntersectType.Disjoint) then
    print("Segment intersects plane")
end
```

See Also

- Intersect.RayPlane3d
- Intersect.LinePlane3d

Intersect.SegmentSegment2d

Brief

Intersect two line segments in 2D.

Definition

```
boolean [, number, number] Intersect.SegmentSegment2d(Vector4 seg1Start, Vector4 seg1Vec, Vector4
seg2Start, Vector4 seg2Vec, boolean ip = false)
boolean [, number, number] Intersect.SegmentSegment2d(number seg1StartX, number seg1StartY, number
seg1VecX, number seg1VecY,
                                                      number seg2StartX, number seg2StartX, number
seg2VecX, number seg2VecY, boolean ip = false)
```

Arguments

seg1Start - first (start) point of segment 1.seg1Vec - vector from first (start) point of segment 1 to second (end) point of segment 1. If the segment points
are A and B, the vector is B-A.seg2Start - first (start) point of segment 2.seg2Vec - vector from first (start) point of segment 2 to second (end) point of segment 2. If the segment points
are A and B, the vector is B-A.seg1StartX - X coordinate of segment 1 start point.seg1StartY - Y coordinate of segment 1 start point.seg1VecX - X

coordinate of segment 1 direction vector (see above).seg1VecY - Y coordinate of segment 1 direction vector (see above).seg2StartX - X coordinate of segment 2 start point.seg2StartY - Y coordinate of segment 2 start point.seg2VecX - X coordinate of segment 2 direction vector (see above).seg2VecY - Y coordinate of segment 2 direction vector (see above).ip - true to return intersection distances, false if not required.

Return Values

True if the segments intersect, false otherwise. If ip is specified, additionally returns the intersection distance for both segments.

Description

Intersect two segments in 2D, and if requested return the intersection distance for both segments if they intersect. Requesting the intersection distances has no additional computation cost for this function. If the intersection point coordinates are required, it can be computed with segStart + t*segVec, where t is the intersection distance for that segment.

Examples

```
seg1Start = Vector4.Create(1, 1)
seg1Vec = Vector4.Create(1, 0)
seg2Start = Vector4.Create(3, 0)
seg2Vec = Vector4.Create(0, 1)
intersect, t1, t2 = Intersect.SegmentSegment2d(seg1Start, seg1Vec, seg2Start, seg2Vec, true)
if (intersect) then
    ip = seg1Start + t1 * seg1Vec       -- or seg2Start + t2 * seg2Vec
end
```

See Also

- Intersect.RayRay2d
- Intersect.RayLine2d
- Intersect.LineSegment2d
- Intersect.LineLine2d
- Intersect.RaySegment2d

Intersect.SegmentSphere3d

Brief

Intersect a segment with a sphere in 3D.

Definition

```
boolean [, number, number] Intersect.SegmentSphere3d(Vector4 segStart, Vector4 segVec, Vector4 center,
number radius, boolean ip = false)
```

Arguments

segStart - start point of segment.segVec - segment vector.center - center point of sphere.radius - radius of sphere.ip - if true, compute and return intersection points.

Return Values

True if the segment intersects the sphere, false otherwise. If ip is specified, the intersection distance along the segment is returned as the second return value.

Description

Intersect a segment with a sphere in 3D, and if requested return the distance along the segment of the intersection point. If "ip" is true the intersection distance will be returned after the boolean return value. If the intersection distance is not requested it will not be computed, which will be more efficient. Of the two forms of this function, the latter (specifying normalized segment direction and length) is more efficient.

Examples

```
segStart = Vector4.Create(0, 1, 0)
segVec = Vector4.Create(6, 0, 0)
center = Vector4.Create(3, 1, 0)
intersect, t1, t2 = Intersect.SegmentSphere3d(segStart, segVec, center, 2, true)
if (intersect) then
    -- t1 must be valid
    print("First intersection at", t1)
    -- t2 may be nil (segment starts inside sphere or doesn't cross sphere), may also be equal to t1
if segment intersects tangentially
    if (t2 ~= nil and t2 ~= t1) then
        print("Second intersection at", t2)
    end
end
```

See Also

- Intersect.RaySphere3d
- Intersect.LineSphere3d

Intersect.SphereAabb3d

Brief

Intersect a sphere and an AABB in 3D.

Definition

```
boolean Intersect.SphereAabb3d(Vector4 center, number radius, Vector4 boxMin, Vector4 boxMax)
```

Arguments

center - Sphere center point.radius - Sphere radius.boxMin - Minimum extents of AABB.boxMax - Maximum extents of AABB.

Return Values

True if the sphere and AABB overlap, or false if not.

Description

Intersect a sphere and an AABB in 3D.

Examples

```
center = Vector4.Create(0, 1, 0)
radius = 2
boxMin = Vector4.Create(1, 0.5, 0)
boxMax = Vector4.Create(3, 1.5, 0)
if (Intersect.CircleSphere3d(center, radius, boxMin, boxMax)) then
    print("Intersection!")
end
```

See Also

None

Intersect.SpherePlane3d

Brief

Intersect a sphere with a plane in 3D.

Definition

```
number Intersect.SpherePlane3d(Vector4 center, number radius, Vector4 plane)
```

Arguments

center - center point of sphere.radius - radius of sphere.plane - plane equation [normal, distance]

Return Values

Penetration distance of sphere, positive if sphere intersects or lies behind plane, negative otherwise.

Description

Intersect a sphere with a plane in 3D and return the penetration distance, which is positive if the sphere intersects or lies behind the plane, or negative otherwise. If 0 <= penetration distance <= 2*radius, the sphere overlaps the plane.

Examples

```
local pen = Intersect.SpherePlane3d(center, radius, plane)
if (pen < 0) then
    print("Sphere lies wholly in the positive halfspace of the plane (no intersection)")
else if (pen <= 2*radius)
    print("Sphere overlaps the plane (0 <= penetration distance <= diameter)")
else
    print("Sphere lies wholly in the negative halfspace of the plane (no intersection)")
end
```

See Also

None

Intersect.SphereSphere3d

Brief

Intersect two spheres in 3D.

Definition

```
boolean [, number] Intersect.SphereSphere3d(Vector4 center1, number radius1, Vector4 center2, number
radius2, boolean pd = false)
```

Arguments

center1 - center point of sphere 1.radius1 - radius of sphere 1.center2 - center point of sphere 2.radius2 - radius of sphere 2.pd - return penetration distance.

Return Values

True if spheres intersect or false otherwise. If pd is true, additionally returns the penetration distance of the spheres.

Description

Intersect two spheres in 2D. If pd is true, also returns the penetration distance, which is >= 0 if the spheres intersect, or negative otherwise. If the penetration distance is greater than the smaller radius of the two, the small sphere lies wholly inside the large one. The penetration distance is defined along the vector between the spheres' centers.

Examples

```
local intersect, pd = Intersect.SphereSphere3d(center1, radius1, center2, radius2, true)
if (intersect) then
    if (pd >= math.min(radius1, radius2)) then
        print("Big sphere wholly contains smaller sphere")
    else
        print("Spheres intersect with penetration distance", pd)
    end
end
```

See Also

None

Json

Json.Decode

Brief

Decode a JSON structure (held in a memory container) into a table.

Definition

```
Json.Decode( MemoryContainer in, number offset = -1, size = -1 )
```

Arguments

in - The memory container holding the JSON-formatted block.offset - Offset into the container where the JSON structure begins.size - The size of the JSON structure.

Return Values

A table containing the decoded data structure, or a negative number on error.
-1 - Invalid argument.
-2 - Parse error.

Description

This function parses a JSON-formatted data structure held in the provided memory container, and returns a corresponding Lua table. Arrays will be treated as key-value pairs with the array index as the key (the base index will be 1).

Examples

```
container = MemoryContainer.Create( 1024 )
```

See Also

- Json.DecodeFromString
- Json.Encode

Json.DecodeFromString

Brief

Decode a JSON structure (held in a string) into a table.

Definition

```
Json.DecodeFromString( string in )
```

Arguments

in - A string holding a JSON-formatted data structure.

Return Values

A table containing the decoded data structure, or a negative number on error.
-1 - Invalid argument.
-2 - Parse error.

Description

This function parses a JSON-formatted data structure and returns a corresponding Lua table. Arrays will be treated as key-value pairs with the array index as the key (the base index will be 1).

Examples

```
decoded = Json.DecodeFromString(
"{\"session\":{\"@expires\":\"1314268686648\",\"@id\":\"3a50cfb2\",\"person\":{\"@id\":\"e8a41543\",\"dis
)
```

See Also

- Json.Decode
- Json.Encode

Json.Encode

Brief

Encode a table into JSON format, storing the results in the provided memory container.

Definition

```
number Json.Encode( MemoryContainer out, table in )
```

Arguments

out - The memory container to write the JSON-encoded data into.in - The table to be encoded.

Return Values

The number of bytes written to the output memory container, or a negative number on error.
-1 - Invalid argument.
-2 - Not enough space in the memory container.
-3 - Unable to encode table in the JSON format.

Description

Write out a JSON-formatted data structure representing the given table to the provided memory container. The structure of the table must conform to the JSON specification.

Examples

```
testtable =
{"session":{"service":{"@id":"bb88aea9","display_name":"Testing"},"person":{"@id":"e8a41543","display_nam
= MemoryContainer.Create( 1024 )
result = Json.Encode( container, testtable )
```

See Also

- Json.Decode
- Json.DecodeFromString

Light

Light.Create

Brief

Creates a new Light.

Definition

```
Light.Create( string lightType )
```

Arguments

lightType - The type of light. Can be one of the following:
"spot"
"point"

Return Values

A new Light instance.

Description

Creates a new Light. The light can be created as a point light or a spot light. Point lights apply light to the scene in a sphere centered around the light's position. Spot lights apply light to the scene in a frustum (pyramidal) shape. On creation, spot lights point down the positive Z axis. The orientation and position of a light can be modified by setting the position and rotation of the light's entity. The light's color defaults to white (red = 1.0, green = 1.0, blue = 1.0). By default, the light will start to attenuate from the position of the entity to which it is attached. Additionally, its contribution will attenuate to zero 2.5m from the position of the entity to which it is attached. The default attenuation power for the light is 2.0. If the light is a spot light, the default field of view angle on creation is 45 degrees.

Examples

```
-- Create a spot light
spotLight = Light.Create( "spot" )
Entity.SetLight(spotEntity, spotLight)

-- Create a point light
pointLight = Light.Create( "point" )
Entity.SetLight(pointEntity, pointLight)
```

See Also

- Entity.SetLight

Light.SetAttenuationEnd

Brief

Sets the distance to the end of the light's attenuation.

Definition

```
Light.SetAttenuationEnd( Light light, number attenuationEnd )
```

Arguments

light - The light to update.attenuationEnd - The distance to the end of the attenuation (in meters).

Return Values

None.

Description

Sets the distance to the end of the light's attenuation. At this distance, the light's contribution falls to zero.

Examples

```
spotLight = Light.Create( "spot" )
Light.SetAttenuationStart( spotLight, 2 )

-- Attenuate the light to zero brightness 5m from its position
Light.SetAttenuationEnd( spotLight, 5 )
```

See Also

- Light.Create
- Light.SetAttenuationPower
- Light.SetAttenuationStart

Light.SetAttenuationPower

Brief

Sets the light's attenuation power.

Definition

```
Light.SetAttenuationPower( Light light, number attenuationPower )
```

Arguments

light - The light to update.attenuationPower - The attenuation power.

Return Values

None.

Description

Sets the light's attenuation power. By default, a newly created light with have an attenuation power of 2.0. The power of the light affects the attenuation factor of the light as follows: attenFactor = (1.0 - saturate((distanceFromLight - attenStart) / (attenEnd - attenStart))) ^ power

Examples

```
pointLight = Light.Create( "point" )
Light.SetAttenuationStart( pointLight, 2 )
Light.SetAttenuationEnd( pointLight, 5 )
Light.SetAttenuationPower( pointLight, 4 )
```

See Also

- Light.Create
- Light.SetAttenuationEnd
- Light.SetAttenuationStart

Light.SetAttenuationStart

Brief

Sets the distance to the start of the light's attenuation.

Definition

```
Light.SetAttenuationStart( Light light, number attenuationStart )
```

Arguments

light - The light to update.attenuationStart - The distance to the start of the attenuation (in meters).

Return Values

None.

Description

Sets the distance to the start of the light's attenuation. From the position of the light to the attenuation start distance, the light's contribution is at its maximum. Its contribution then falls to zero between the attenuation start and end distances.

Examples

```
spotLight = Light.Create( "spot" )

-- Attenuate the light from full brightness 2m from its position
Light.SetAttenuationStart( spotLight, 2 )
Light.SetAttenuationEnd( spotLight, 5 )
```

See Also

- Light.Create
- Light.SetAttenuationPower
- Light.SetAttenuationEnd

Light.SetColor

Brief

Set the color of the light.

Definition

```
Light.SetColor( Light light, Vector4 rgb )
```

Arguments

light - The light to update.rgb - The color to set the light, stored in red, green, blue order.

Return Values

None.

Description

Set the color of the light. The light color is not clamped and can go above 1.0. The alpha component of the Vector4 is ignored and need not be specified.

Examples

```
spotLight = Light.Create( "spot" )
Light.SetColor( spotLight, Vector4.Create( 4.0, 3.0, 2.0 ) )
```

See Also

- Light.Create

Light.SetMask

Brief

Sets the mask texture for the light.

Definition

```
Light.SetMask( Light light, Texture mask )
```

Arguments

light - The light to update.mask - The mask texture for the light.

Return Values

None.

Description

Sets the mask texture for the light. Spot lights project a quad of light onto the scene. This quad can be masked with the supplied mask texture. The lighter the pixel in the mask, the more light will pass through it. For example, a white circle on a black background will project a circle onto the environment that is the color of the spot light. The mask for a spot light should be a DXT1 DDS texture. Point lights apply light all around them. Therefore, to mask the light of a point light, specify a cube map texture. Once again, this should be in DXT1 DDS format.

Examples

```
-- Create a masked spot light
spotLight = Light.Create( "spot" )
Light.SetMask( spotLight, Texture.Find( "circle_mask" )

-- Create a masked spot light
pointLight = Light.Create( "point" )
Light.SetMask( pointLight, Texture.Find( "cube_map_mask" )
```

See Also

- Light.Create
- Texture.Find

Light.SetShadowQuality

Brief

Set the quality of the dynamic shadows.

Definition

```
Light.SetShadowQuality( Light light, ShadowQuality shadowQuality )
```

Arguments

light - The light to update.shadowQuality - A ShadowQuality enum value:
ShadowQuality.Low
ShadowQuality.Medium
ShadowQuality.High

Return Values

None.

Description

Set the quality of dynamic shadows cast by this light. Higher quality shadows are more expensive to render. The default shadow quality is low.

Examples

```
spotLight = Light.Create( "spot" )
Light.SetSpotShadow( spotLight, true )
Light.SetShadowQuality( spotLight, ShadowQuality.Low )
```

See Also

- Light.Create
- Light.SetMask

Light.SetSpotFov

Brief

Sets the field of view of a spot light.

Definition

```
Light.SetSpotFov( Light light, number fov )
```

Arguments

light - The light to update.fov - The field of view of the spot in degrees.

Return Values

None.

Description

Sets the field of view of a spot light's frustum in degrees. This function is only valid for spot lights. The field of view of a newly created light defaults to 45 degrees.

Examples

```
spotLight = Light.Create( "spot" )
Light.SetSpotFov( spotLight, 90 )
```

See Also

- Light.Create

Light.SetSpotShadow

Brief

Set the spot light to casts shadows or not.

Definition

```
Light.SetSpotShadow( Light light, boolean useShadow )
```

Arguments

light - The light to update.useShadow - If true the spot light casts shadows of objects obstructing it and if false it casts no shadows.

Return Values

None.

Description

If true, the spot light casts shadows of objects obstructing it. If false, the spot light casts no shadows. By default, a newly created spot light casts no shadows.

Examples

```
spotLight = Light.Create( "spot" )
Light.SetSpotShadow( spotLight, true )
```

See Also

- Light.Create
- Light.SetMask

LinearCast

LinearCast.DebugDraw

Brief

Render a debug visual of the linear cast.

Definition

```
LinearCast.DebugDraw(LinearCast cast, Vector4 color)
```

Arguments

cast - cast to draw.color - color to render with.

Return Values

None.

Description

A debug representation of the linear cast will be drawn. Because the swept shape may be quite complex, this function will render the cast collision shape at the begin and end points of the cast and draw a line between the two points. This function requires the debug flag to be set on your object. The render will only persist for one frame, so you will need to call this function every frame while you wish to display the visualization.

Examples

```
cast:DebugDraw(Vector.Create(0, 1, 0, 1))       -- draw in green
```

See Also

- Collision.CreateLinearCast

LinearCast.GetEndPoint

Brief

Get the end point of a linear cast.

Definition

```
LinearCast.GetEndPoint(LinearCast cast, Vector4 point)
```

Arguments

cast - cast to query.point (out) - Vector4 to receive the point.

Return Values

None.

Description

Get the end point of a linear cast.

Examples

```
local v = Vector4.Create()
cast:GetStartPoint(v)
print("Cast start point is:", v)
```

See Also

- LinearCast.GetStartPoint
- LinearCast.GetHitPoint
- LinearCast.IsReady

LinearCast.GetHitCount

Brief

Retrieve the number of objects hit by the linear cast.

Definition

```
number LinearCast.GetHitCount(LinearCast cast)
```

Arguments

cast - the cast to query.

Return Values

The number of hits.

Description

The number of hits will be between 0 and the maxOverlaps value passed to the Collision.CreateLinearCast call.

Examples

```
cast = Collision.CreateLinearCast(ShapeType.Box, Vector4.Create(0.2, 0.8, 0.2), 5)      -- create cast
of box of half-extents [0.2, 0.8, 0.2], 5 max overlaps
rot = Matrix44.Create()
rot:SetRotationXyz(0, 45, 0)
cast:SetRotation(rot)
cast:Start(Vector4.Create(0,0,0), Vector4.Create(5,0,0))
while (not cast:IsReady()) do
    coroutine.yield()
end
local v = Vector4.Create()
for i = 1, cast:GetHitCount() do
    cast:GetHitPoint(i, v)
    print("Hit point:", v)
    cast:GetHitNormal(i, v)
    print("Hit normal:", v)
    local type = cast:GetHitType(i)
    print("Hit type:", type)
    print("Hit distance:", cast:GetHitDistance(i))
    local data = cast:GetHitData(i)
    if (type == HitType.Entity) then
        -- data is an entity
    elseif (type == HitType.Person) then
        -- data is a person
    elseif (type == HitType.Furniture) then
        -- data is a table (see GetHitData)
    elseif (type == HitType.Scene) then
        -- data is user ID of scene feature hit
    else
        -- data is nil - no further info available
    end
end
```

See Also

- LinearCast.GetHitPoint
- LinearCast.GetHitNormal
- LinearCast.GetHitType
- LinearCast.GetHitData
- LinearCast.GetHitDistance

LinearCast.GetHitData

Brief

Retrieve additional data about the object hit by a linear cast.

Definition

```
object LinearCast.GetHitData(LinearCast cast, number index)
```

Arguments

cast - the cast to query.index - the index of the hit to query.

Return Values

If a script entity owned by this object is hit, the Entity will be returned; if the entity is not owned by the current object, nil will be returned. If the local avatar is

hit, the Person object representing the local person will be hit. If an item of furniture is hit,
a table of information about the furniture item is returned; see Furniture.GetItemData for info. For world collisions, the surface
user ID will be returned, or 0 if there is none.

Description

Depending on the type of object hit (see LinearCast.GetHitType), this function will access the hit object or further information about the hit object. Note that if an entity owned by another object is hit, this function will return nil.

Examples

```
cast = Collision.CreateLinearCast(ShapeType.Box, Vector4.Create(0.2, 0.8, 0.2), 5)      -- create cast
of box of half-extents [0.2, 0.8, 0.2], 5 max overlaps
rot = Matrix44.Create()
rot:SetRotationXyz(0, 45, 0)
cast:SetRotation(rot)
cast:Start(Vector4.Create(0,0,0), Vector4.Create(5,0,0))
while (not cast:IsReady()) do
    coroutine.yield()
end
local v = Vector4.Create()
for i = 1, cast:GetHitCount() do
    cast:GetHitPoint(i, v)
    print("Hit point:", v)
    cast:GetHitNormal(i, v)
    print("Hit normal:", v)
    local type = cast:GetHitType(i)
    print("Hit type:", type)
    print("Hit distance:", cast:GetHitDistance(i))
    local data = cast:GetHitData(i)
    if (type == HitType.Entity) then
        -- data is an entity
    elseif (type == HitType.Person) then
        -- data is a person
    elseif (type == HitType.Furniture) then
        -- data is a table (see GetHitData)
    elseif (type == HitType.Scene) then
        -- data is user ID of scene feature hit
    else
        -- data is nil - no further info available
    end
end
```

See Also

- LinearCast.GetHitPoint
- LinearCast.GetHitNormal
- LinearCast.GetHitType
- LinearCast.GetHitData
- LinearCast.GetHitCount
- Furniture.GetItemData

LinearCast.GetHitDistance

Brief

Retrieve the distance along the linear cast at which a specified hit occurred.

Definition

```
number LinearCast.GetHitDistance(LinearCast cast, number index)
```

Arguments

cast - the cast to query.index - the index of the hit to query.

Return Values

The distance along the cast, as a proportion of the total distance of the cast [0 to 1].

Description

Returns the TOI (time of impact) along the linear cast path for a specified hit. The distance is in the range [0, 1], and can be converted to a point along the path by computing startPoint + (endPoint-startPoint)*distance. The contact will always appear on the side of the colliding object facing the cast, which may cause confusion if the cast is initially intersecting the object, as the contact distance will be 0 but the contact point may lie "behind" the origin of the cast. Scripts should be sure to check that the hit distance makes sense for their use.

Examples

```
cast = Collision.CreateLinearCast(ShapeType.Box, Vector4.Create(0.2, 0.8, 0.2), 5)      -- create cast
of box of half-extents [ 0.2, 0.8, 0.2 ], 5 max overlaps
rot = Matrix44.Create()
rot:SetRotationXyz(0, 45, 0)
cast:SetRotation(rot)
cast:Start(Vector4.Create(0,0,0), Vector4.Create(5,0,0))
while (not cast:IsReady()) do
    coroutine.yield()
end
local v = Vector4.Create()
for i = 1, cast:GetHitCount() do
    cast:GetHitPoint(i, v)
    print("Hit point:", v)
    cast:GetHitNormal(i, v)
    print("Hit normal:", v)
    local type = cast:GetHitType(i)
    print("Hit type:", type)
    print("Hit distance:", cast:GetHitDistance(i))
    local data = cast:GetHitData(i)
    if (type == HitType.Entity) then
        -- data is an entity
    elseif (type == HitType.Person) then
        -- data is a person
    elseif (type == HitType.Furniture) then
        -- data is a table (see GetHitData)
    elseif (type == HitType.Scene) then
        -- data is user ID of scene feature hit
    else
        -- data is nil - no further info available
    end
end
```

See Also

- LinearCast.GetHitPoint
- LinearCast.GetHitNormal
- LinearCast.GetHitType
- LinearCast.GetHitData
- LinearCast.GetHitCount

LinearCast.GetHitNormal

Brief

Retrieve the normal of the contact point generated for a specified hit.

Definition

```
number LinearCast.GetHitNormal(LinearCast cast, number index, Vector4 normal)
```

Arguments

cast - the cast to query.index - the index of the hit to query.normal (out) - the Vector4 to receive the hit normal.

Return Values

None.

Description

Retrieves the normal of the contact point generated for the specified hit. The contact normal will depend on the shapes of the cast and the colliding object. The contact will always appear on the side of the colliding object facing the cast, which may cause confusion if the cast is initially intersecting the object, as the contact point will be "behind" the start of the cast. Scripts should take care to examine hit normals and ensure they are suitable for the script's purpose.

Examples

```
cast = Collision.CreateLinearCast(ShapeType.Box, Vector4.Create(0.2, 0.8, 0.2), 5)      -- create cast
of box of half-extents [0.2, 0.8, 0.2], 5 max overlaps
rot = Matrix44.Create()
rot:SetRotationXyz(0, 45, 0)
cast:SetRotation(rot)
cast:Start(Vector4.Create(0,0,0), Vector4.Create(5,0,0))
while (not cast:IsReady()) do
    coroutine.yield()
end
local v = Vector4.Create()
for i = 1, cast:GetHitCount() do
    cast:GetHitPoint(i, v)
    print("Hit point:", v)
    cast:GetHitNormal(i, v)
    print("Hit normal:", v)
    local type = cast:GetHitType(i)
    print("Hit type:", type)
    print("Hit distance:", cast:GetHitDistance(i))
    local data = cast:GetHitData(i)
    if (type == HitType.Entity) then
        -- data is an entity
    elseif (type == HitType.Person) then
        -- data is a person
    elseif (type == HitType.Furniture) then
        -- data is a table (see GetHitData)
    elseif (type == HitType.Scene) then
        -- data is user ID of scene feature hit
    else
        -- data is nil - no further info available
    end
end
```

See Also

- LinearCast.GetHitPoint
- LinearCast.GetHitDistance
- LinearCast.GetHitType

- LinearCast.GetHitData
- LinearCast.GetHitCount

LinearCast.GetHitPoint

Brief

Retrieve the contact point generated for a specified hit.

Definition

```
LinearCast.GetHitPoint(LinearCast cast, number index, Vector4 point)
```

Arguments

cast - the cast to query.index - the index of the hit to query.point (out) - the Vector4 to receive the hit point.

Return Values

None.

Description

Retrieves the position of the contact point generated for the specified hit. The contact point will depend on the shapes of the cast and the colliding object. The contact will always appear on the side of the colliding object facing the cast, which may cause confusion if the cast is initially intersecting the object, as the contact point will be "behind" the start of the cast. Scripts should take care to examine hit points and ensure they are suitable for the script's purpose.

Examples

```
cast = Collision.CreateLinearCast(ShapeType.Box, Vector4.Create(0.2, 0.8, 0.2), 5)      -- create cast
of box of half-extents [0.2, 0.8, 0.2], 5 max overlaps
rot = Matrix44.Create()
rot:SetRotationXyz(0, 45, 0)
cast:SetRotation(rot)
cast:Start(Vector4.Create(0,0,0), Vector4.Create(5,0,0))
while (not cast:IsReady()) do
    coroutine.yield()
end
local v = Vector4.Create()
for i = 1, cast:GetHitCount() do
    cast:GetHitPoint(i, v)
    print("Hit point:", v)
    cast:GetHitNormal(i, v)
    print("Hit normal:", v)
    local type = cast:GetHitType(i)
    print("Hit type:", type)
    print("Hit distance:", cast:GetHitDistance(i))
    local data = cast:GetHitData(i)
    if (type == HitType.Entity) then
        -- data is an entity
    elseif (type == HitType.Person) then
        -- data is a person
    elseif (type == HitType.Furniture) then
        -- data is a table (see GetHitData)
    elseif (type == HitType.Scene) then
        -- data is user ID of scene feature hit
    else
        -- data is nil - no further info available
    end
end
```

See Also

- LinearCast.GetHitDistance
- LinearCast.GetHitNormal
- LinearCast.GetHitType
- LinearCast.GetHitData
- LinearCast.GetHitCount

LinearCast.GetHitType

Brief

Retrieve the type of the object hit by the cast.

Definition

```
HitType LinearCast.GetHitType(LinearCast cast, number index)
```

Arguments

cast - the cast to query.index - the index of the hit to query.

Return Values

The type of the colliding object.

Description

There are several types of colliding objects in most scenes, including script entities, the local avatar, furniture and the scene itself. This function will indicate which type of object the cast has hit. For script entities, the local avatar and furniture, further information is available for the collided object; see LinearCast.GetHitData.

Examples

```
cast = Collision.CreateLinearCast(ShapeType.Box, Vector4.Create(0.2, 0.8, 0.2), 5)      -- create cast
of box of half-extents [0.2, 0.8, 0.2], 5 max overlaps
rot = Matrix44.Create()
rot:SetRotationXyz(0, 45, 0)
cast:SetRotation(rot)
cast:Start(Vector4.Create(0,0,0), Vector4.Create(5,0,0))
while (not cast:IsReady()) do
    coroutine.yield()
end
local v = Vector4.Create()
for i = 1, cast:GetHitCount() do
    cast:GetHitPoint(i, v)
    print("Hit point:", v)
    cast:GetHitNormal(i, v)
    print("Hit normal:", v)
    local type = cast:GetHitType(i)
    print("Hit type:", type)
    print("Hit distance:", cast:GetHitDistance(i))
    local data = cast:GetHitData(i)
    if (type == HitType.Entity) then
        -- data is an entity
    elseif (type == HitType.Person) then
        -- data is a person
    elseif (type == HitType.Furniture) then
        -- data is a table (see GetHitData)
    elseif (type == HitType.Scene) then
        -- data is user ID of scene feature hit
    else
        -- data is nil - no further info available
    end
end
```

See Also

- LinearCast.GetHitPoint
- LinearCast.GetHitNormal
- LinearCast.GetHitDistance
- LinearCast.GetHitData
- LinearCast.GetHitCount

LinearCast.GetStartPoint

Brief

Get the start point of a linear cast.

Definition

```
LinearCast.GetStartPoint(LinearCast cast, Vector4 point)
```

Arguments

cast - cast to query.point (out) - Vector4 to receive the point.

Return Values

None.

Description

Get the start point of a linear cast.

Examples

```
local v = Vector4.Create()
cast:GetStartPoint(v)
print("Cast start point is:", v)
```

See Also

- LinearCast.GetEndPoint
- LinearCast.Start

LinearCast.IsReady

Brief

Query whether a linear cast is complete.

Definition

```
boolean LinearCast.IsReady(LinearCast cast)
```

Arguments

cast - cast to query.

Return Values

True if the cast is complete, false otherwise.

Description

If the cast has not been started, this function will return false. No result retrieval functions will work if this function returns false.

Examples

```
cast = Collision.CreateLinearCast(ShapeType.Box, Vector4.Create(0.2, 0.8, 0.2), 5)      -- create cast
of box of half-extents [0.2, 0.8, 0.2], 5 max overlaps
rot = Matrix44.Create()
rot:SetRotationXyz(0, 45, 0)
cast:SetRotation(rot)
cast:Start(Vector4.Create(0,0,0), Vector4.Create(5,0,0))
while (not cast:IsReady()) do
    coroutine.yield()
end
local v = Vector4.Create()
for i = 1, cast:GetHitCount() do
    cast:GetHitPoint(i, v)
    print("Hit point:", v)
    cast:GetHitNormal(i, v)
    print("Hit normal:", v)
    local type = cast:GetHitType(i)
    print("Hit type:", type)
    print("Hit distance:", cast:GetHitDistance(i))
    local data = cast:GetHitData(i)
    if (type == HitType.Entity) then
        -- data is an entity
    elseif (type == HitType.Person) then
        -- data is a person
    elseif (type == HitType.Furniture) then
        -- data is a table (see GetHitData)
    elseif (type == HitType.Scene) then
        -- data is user ID of scene feature hit
    else
        -- data is nil - no further info available
    end
end
```

See Also

- LinearCast.Start

LinearCast.SetCollisionLayersToHit

Brief

Specify which collision layers the cast should hit.

Definition

```
LinearCast.SetCollisionLayersToHit(LinearCast raycast, ...)
```

Arguments

cast - the cast to affect.... - collision layer(s) to hit, specified as separate arguments. Pass no args to clear collision filter.

Return Values

None.

Description

Specify which collision layers the cast should hit. The collision layers should be specified using the CollisionLayer enums. A call to this function will override all previous calls this function. If collision layers are specified, the user should not use the 'collisionLayer' parameter of Raycast.Start. If no collision filter is specified, the cast will revert to standard behaviour and the user may use the 'collisionLayer' parameter of LinearCast.Start.

Examples

```
cast:SetCollisionLayersToHit(CollisionLayer.Avatar, CollisionLayer.Entity)
-- to clear, cast:SetCollisionLayersToHit()
```

See Also

- LinearCast.Start

LinearCast.SetRotation

Brief

Set the rotation on the collision shape of a linear cast.

Definition

```
LinearCast.SetRotation(LinearCast cast, Matrix44 rotation)
```

Arguments

cast - cast to modify.rotation - rotation matrix to use (top-left 3x3 submatrix used only).

Return Values

None.

Description

The collision shape used for the sweep through the world can be rotated if required. The rotation is fixed, so the same orientation is used for the whole sweep (the shape cannot rotate during the sweep). Only the top-left 3x3 portion of the matrix is used. Scale is not supported but will not be validated against, so be sure to not introduce any scale.

Examples

```
cast = Collision.CreateLinearCast(ShapeType.Box, Vector4.Create(0.2, 0.8, 0.2), 5)      -- create cast
of box of half-extents [0.2, 0.8, 0.2], 5 max overlaps
rot = Matrix44.Create()
rot:SetRotationXyz(0, 45, 0)
cast:SetRotation(rot)
cast:Start(Vector4.Create(0,0,0), Vector4.Create(5,0,0))
while (not cast:IsReady()) do
    coroutine.yield()
end
local v = Vector4.Create()
for i = 1, cast:GetHitCount() do
    cast:GetHitPoint(i, v)
    print("Hit point:", v)
    cast:GetHitNormal(i, v)
    print("Hit normal:", v)
    local type = cast:GetHitType(i)
    print("Hit type:", type)
    print("Hit distance:", cast:GetHitDistance(i))
    local data = cast:GetHitData(i)
    if (type == HitType.Entity) then
        -- data is an entity
    elseif (type == HitType.Person) then
        -- data is a person
    elseif (type == HitType.Furniture) then
        -- data is a table (see GetHitData)
    elseif (type == HitType.Scene) then
        -- data is user ID of scene feature hit
    else
        -- data is nil - no further info available
    end
end
```

See Also

- LinearCast.Start

LinearCast.Start

Brief

Perform a linear cast between two points.

Definition

```
LinearCast.Start(LinearCast cast, Vector4 start, Vector4 end, CollisionLayer layer =
CollisionLayer.Cast, Phys3dWorld world = nil)
```

Arguments

start - Start point.end - End point.layer - Collision layer of the cast. Must specify nil here if using the collision filter.world - Phys3dWorld to sweep through, or nil for the default world.

Return Values

None.

Description

Starts the processing of a linear cast. The results will not be available until LinearCast.IsReady returns true. Note that to set the orientation of the cast shape, use SetRotation before calling Start. The orientation will be fixed over the cast.

Examples

```
cast = Collision.CreateLinearCast(ShapeType.Box, Vector4.Create(0.2, 0.8, 0.2), 5)      -- create cast
of box of half-extents [0.2, 0.8, 0.2], 5 max overlaps
rot = Matrix44.Create()
rot:SetRotationXyz(0, 45, 0)
cast:SetRotation(rot)
cast:Start(Vector4.Create(0,0,0), Vector4.Create(5,0,0))
while (not cast:IsReady()) do
    coroutine.yield()
end
local v = Vector4.Create()
for i = 1, cast:GetHitCount() do
    cast:GetHitPoint(i, v)
    print("Hit point:", v)
    cast:GetHitNormal(i, v)
    print("Hit normal:", v)
    local type = cast:GetHitType(i)
    print("Hit type:", type)
    print("Hit distance:", cast:GetHitDistance(i))
    local data = cast:GetHitData(i)
    if (type == HitType.Entity) then
        -- data is an entity
    elseif (type == HitType.Person) then
        -- data is a person
    elseif (type == HitType.Furniture) then
        -- data is a table (see GetHitData)
    elseif (type == HitType.Scene) then
        -- data is user ID of scene feature hit
    else
        -- data is nil - no further info available
    end
end
```

See Also

- Collision.CreateLinearCast
- LinearCast.SetRotation
- LinearCast.IsReady

LocalPlayer

LocalPlayer.AddRepertoire

Brief

Adds the specified repertoire resource from the object provided, to the local player. The action is replicated across the network, so the remote person will add the repertoire

Definition

```
Repertoire LocalPlayer.AddRepertoire( string label, number priority )
```

Arguments

label - the label of the repertoire to add.
priority - the priority over existing repertoires already added. The higher the number, the greater the priority. The default repertoire is added with a priority of zero.

Return Values

Reference to the repertoire resource that was added, or nil if failed to add.

Description

Examples

```
repertoire = LocalPlayer.AddRepertoire( "level1", 0 )
```

See Also

None

LocalPlayer.AddToPlayersMetList

Brief

Add a player to the XMB Players Met List.

Definition

```
LocalPlayer.AddToPlayersMetList( string playerName )
```

Arguments

personId - PSN ID of the player to add.

Return Values

None.

Description

The player won't be added if they're a Friend, or Blocked, or if they're the Local Player. Also, the player won't be added if they're not currently in the scene. It's advised this function is used sparingly and with care. E.g. if used to add other players in a Mini-game, please limit it to 16.

Examples

```
LocalPlayer.AddToPlayersMetList("MyPSNId")
```

See Also

None

LocalPlayer.DebugRemovePortable

Brief

Removes the local player's person's active object.

Definition

```
LocalPlayer.DebugRemoveActiveObject( string objectId )
```

Arguments

None

Return Values

None.

Description

Removes the local player's person's active object. For debug purposes only.

Examples

```
LocalPlayer.DebugRemoveActiveObject()
```

See Also

None

LocalPlayer.DebugSetPortable

Brief

Sets the local player's portable.

Definition

```
LocalPlayer.DebugSetPortable( string objectId )
```

Arguments

objectId - The object ID of the portable.

Return Values

None.

Description

Sets the local player's portable. For debug purposes only.

Examples

```
LocalPlayer.DebugSetPortable( "00000000-00000000-00000000-00000001" )
```

See Also

None

LocalPlayer.DoAction

Brief

Makes the local player's avatar perform the specified action.

Definition

```
LocalPlayer.DoAction( string actionName )
```

Arguments

actionName - Name of the action to be performed. It can be a custom animation loaded through LocalPlayer.AddRepertoire as well as one of the following standard actions:
"agree"
"beckon"
"bow"
"cheer"
"clap"
"disagree"
"frustration"
"laugh"
"point"
"wave"
"stand_up_from_seat"

Return Values

true if the action was recognised, false otherwise.

Description

Makes the local player's avatar perform the specified action.

Examples

```
LocalPlayer.DoAction( "wave" )
```

See Also

- LocalPlayer.IsDoingAction
- LocalPlayer.AddRepertoire

LocalPlayer.EnableAutoHiding

Brief

Enable/Disable the automatic hiding of the LocalPlayer's avatar when the camera gets too close.

Definition

```
LocalPlayer.EnableAutoHiding( boolean enable )
```

Arguments

enable - false to turn off auto-hiding, or true to re-enable it.

Return Values

None.

Description

The client automatically hides the avatar when the camera gets too close, for usability and aesthetic reasons. In certain circumstances it may be advantageous to disable this behavior, such as when creating a first person view.

Examples

```
LocalPlayer.EnableAutoHiding(false)
```

See Also

None

LocalPlayer.GetAccountRegion

Brief

Returns a string representing the PSN account region.

Definition

```
string LocalPlayer.GetAccountRegion()
```

Arguments

None

Return Values

A string representing the PSN account region.

Description

Returns a string representing the PSN account region.

Examples

```
if ( LocalPlayer.GetAccountRegion() == "en-GB" ) then
    print( "I am British!" )
end
```

See Also

- LocalPlayer.GetHomeRegion
- LocalPlayer.GetCurrentRegion

LocalPlayer.GetAccountTerritory

Brief

Returns a string representing the PSN account territory.

Definition

```
string LocalPlayer.GetAccountTerritory()
```

Arguments

None

Return Values

A string representing the PSN account territory.

Description

Returns a string representing the PSN account territory. Can be one of the following:

"SCEE" (Europe)

"SCEA" (North America)

"SCEJ" (Japan)

"SCEAsia" (Rest of Asia)

Examples

```
if ( LocalPlayer.GetAccountTerritory() == "SCEE" ) then
    print( "I am European!" )
end
```

See Also

- LocalPlayer.GetAccountRegion
- System.GetRegionsTerritory

LocalPlayer.GetAge

Brief

Returns the local player's age in years.

Definition

```
number LocalPlayer.GetAge()
```

Arguments

None

Return Values

The local player's age in years.

Description

Returns the local player's age in years. If there are any errors (such as being signed out), this will return -1.

Examples

```
print( 'I am ' .. LocalPlayer.GetAge() .. ' year(s) old.' )
```

See Also

None

LocalPlayer.GetAvatarRig

Brief

Returns the rig type (gender) of the LocalPlayer avatar.

Definition

```
RigType LocalPlayer.GetAvatarRig()
```

Arguments

None

Return Values

The rig type of the LocalPlayer avatar, as a number from the enumerated type RigType (see description).

Description

Returns the rig type (gender) of the LocalPlayer avatar, as one of the following values:
RigType.Undefined
RigType.Male
RigType.Female

Examples

```
if (LocalPlayer.GetAvatarRig() == RigType.Male) then
    LocalPlayer.SetRigComponentObjectId(kMaleHatObjectId)
else
    LocalPlayer.SetRigComponentObjectId(kFemaleHatObjectId)
end
```

See Also

- LocalPlayer.SetAvatarRig

LocalPlayer.GetCurrentLogin

Brief

Returns the date of the current login.

Definition

```
bool LocalPlayer.GetCurrentLogin( Time t )
```

Arguments

t - a time value to fill in.

Return Values

Returns true if the first session date is set and a valid Time object was provided.

Description

This will return the date/time of the current login. Used to determine how long the user has been online. Note that this value is not guaranteed to be tamper-proof as it is obtained from the user's hard disk; the value should be sanity-checked, and if its accuracy is critically important, should be verified with an external source.

Examples

```
t = Time.Create()
if ( LocalPlayer.GetCurrentLogin( t ) )
    print( 'Current login is ' .. Time.GetTime( t ) .. '.' )
```

See Also

- LocalPlayer.GetUserType
- LocalPlayer.GetPreviousUserType
- LocalPlayer.GetFirstLogin
- LocalPlayer.GetLastLogin
- LocalPlayer.GetLoginCount
- Time.Create

LocalPlayer.GetCurrentRegion

Brief

Returns a string representing the region of the space that the local player is currently in.

Definition

```
string LocalPlayer.GetCurrentRegion()
```

Arguments

None

Return Values

A string representing the current region the local player is in.

Description

Returns a string representing the region of the space that the local player is currently in. This is one of the PS Home supported regions.

Examples

```
if ( LocalPlayer.GetCurrentRegion() == "en-US" ) then
    print( "I am in an American space!" )
end
```

See Also

- LocalPlayer.GetHomeRegion
- LocalPlayer.GetAccountRegion

LocalPlayer.GetFirstLogin

Brief

Returns the first login date.

Definition

```
bool LocalPlayer.GetFirstLogin( Time t )
```

Arguments

t - a time value to fill in.

Return Values

Returns true if the first session date is set and a valid Time object was provided.

Description

This will return the date/time of the first login. Login tracking was added for the 1.55 release and therefore this date/time will be dated from this release. Clearing savedata will reset this date. Note that this value is not guaranteed to be tamper-proof as it is obtained from the user's hard disk; the value should be sanity-checked, and if its accuracy is critically important, should be verified with an external source.

Examples

```
t = Time.Create()
if ( LocalPlayer.GetFirstLogin( t ) )
    print( 'First login was ' .. Time.GetTime( t ) .. '.' )
```

See Also

- LocalPlayer.GetUserType
- LocalPlayer.GetPreviousUserType
- LocalPlayer.GetLastLogin
- LocalPlayer.GetCurrentLogin
- LocalPlayer.GetLoginCount
- Time.Create

LocalPlayer.GetFriendCount

Brief

Returns the number of friends.

Definition

```
number LocalPlayer.GetFriendCount()
```

Arguments

None

Return Values

The number of friends.

Description

Returns the number of friends that the local player has.

Examples

```
print( 'I have ' .. LocalPlayer.GetFriendCount() .. ' friends.' )
```

See Also

- LocalPlayer.GetFriends

LocalPlayer.GetFriends

Brief

Returns an array of friends as User objects.

Definition

```
User[] LocalPlayer.GetFriends()
```

Arguments

None

Return Values

An array of User objects

Description

Returns an array of friends as User objects.

Examples

```
friends = LocalPlayer.GetFriends()
for index, friend in ipairs( friends ) do
    print( User.GetName( friend ) )
end
```

See Also

- User.GetName

LocalPlayer.GetHomeAccountId

Brief

Fills in the unique accountId for the player

Definition

```
LocalPlayer.GetLoginCount()
```

Arguments

bigInt - Unique account id to set

Return Values

None.

Description

This 32-bit accountId is allocated by the Home game servers. It is unique and fixed for each account in the current environment. It will not be valid when running offline. The primary use for this function is to acquire a unique 32-bit ID that can be used in the RelocateToUniqueInstance function - allowing you to send the user to a space that is unique to their account.

Examples

```
local myAccountId = BigInt.Create(32)
LocalPlayer.GetHomeAccountId( myAccountId )
```

See Also

None

LocalPlayer.GetHomeRegion

Brief

Returns a string representing the local player's PS Home region.

Definition

```
string LocalPlayer.GetHomeRegion()
```

Arguments

None

Return Values

A string representing the PS Home region.

Description

Returns a string representing the local player's PS Home region. The account region is remapped into a supported PS Home region as PS Home does not support all regions.

Examples

```
if ( LocalPlayer.GetHomeRegion() == "en-GB" ) then
    print( "My PSN account region is mapped to British in PS Home!" )
end
```

See Also

- LocalPlayer.GetHomeRegion
- LocalPlayer.GetCurrentRegion

LocalPlayer.GetLastLogin

Brief

Returns the last login date.

Definition

```
bool LocalPlayer.GetLastLogin( Time t )
```

Arguments

t - a time value to fill in.

Return Values

Returns true if the last session date is set and a valid Time object was provided. Note that this value is not guaranteed to be tamper-proof as it is obtained from the user's hard disk; the value should be sanity-checked, and if its accuracy is critically important, should be verified with an external source.

Description

This will return the date/time of the last login.

Examples

```
t = Time.Create()
if ( LocalPlayer.GetLastLogin( t ) )
    print( 'Lost login was ' .. Time.GetTime( t ) .. '.' )
```

See Also

- LocalPlayer.GetUserType
- LocalPlayer.GetPreviousUserType
- LocalPlayer.GetFirstLogin
- LocalPlayer.GetCurrentLogin
- LocalPlayer.GetLoginCount
- Time.Create

LocalPlayer.GetLoginCount

Brief

Returns the total number of sessions for the user.

Definition

```
number LocalPlayer.GetLoginCount()
```

Arguments

None.

Return Values

number - The number of sessions

Description

The total number of previous sessions the user has logged in. A new user will have a value of 0. Login tracking was added for the 1.55 release and therefore this count will be since this release. Note that this value is not guaranteed to be tamper-proof as it is obtained from the user's hard disk; the value should be sanity-checked, and if its accuracy is critically important, should be verified with an external source. Clearing savedata will reset this count.

Examples

```
print( 'Number of logins: ' .. LocalPlayer.GetLoginCount() )
```

See Also

- LocalPlayer.GetPreviousUserType
- LocalPlayer.GetFirstLogin
- LocalPlayer.GetLastLogin
- LocalPlayer.GetCurrentLogin

LocalPlayer.GetPerson

Brief

Returns the person object of the local player.

Definition

```
Person LocalPlayer.GetPerson()
```

Arguments

None.

Return Values

Returns the person object of the local player.

Description

Returns the person object of the local player.

Examples

```
me = LocalPlayer.GetPerson()
```

See Also

None

LocalPlayer.GetPersonId

Brief

Returns the local player's person ID.

Definition

```
string LocalPlayer.GetPersonId()
```

Arguments

None

Return Values

The local player's person ID.

Description

Returns the local player's person ID. See Person.GetPersonId for more information on person IDs. This will return the same result as LocalPlayer.GetPerson():GetId().

Examples

```
print( 'My person ID is ' ... LocalPlayer.GetPersonId() )
```

See Also

LocalPlayer.GetPreviousUserType

Brief

Returns the user type from the previous login.

Definition

```
UserType LocalPlayer.GetPreviousUserType()
```

Arguments

None.

Return Values

One of the following user types :-
UserType.New - This is a new user
UserType.Returning - An old user who has returned
UserTYpe.Regular - A returning frequent user
UserType.Unknown - An unknown user type

Description

Returns the previous session's user type. This can be used to find out if the user type has just changed.

Examples

```
if ( LocalPlayer.GetUserType() ~= LocalPlayer.GetPrevousUserType() )
    print( 'User type has changed this session' )
```

See Also

- LocalPlayer.GetUserType
- LocalPlayer.GetFirstLogin
- LocalPlayer.GetLastLogin
- LocalPlayer.GetCurrentLogin
- LocalPlayer.GetLoginCount

LocalPlayer.GetUser

Brief

Returns the user object of the local player.

Definition

```
User LocalPlayer.GetUser()
```

Arguments

None

Return Values

Returns the user object of the local player.

Description

Returns the user object of the local player.

Examples

```
me = LocalPlayer.GetUser()
```

See Also

None

LocalPlayer.GetUserType

Brief

Returns the active user type.

Definition

```
UserType LocalPlayer.GetUserType()
```

Arguments

None.

Return Values

One of the following user types :-
UserType.New - This is a new user
UserType.Returning - An old user who has returned
UserTYpe.Regular - A returning frequent user
UserType.Unknown - An unknown user type

Description

Returns the current type of user logged in.

Examples

```
if ( LocalPlayer.GetUserType() == UserType.New )
    print( 'New user' )
```

See Also

- LocalPlayer.GetPreviousUserType

- LocalPlayer.GetFirstLogin
- LocalPlayer.GetLastLogin
- LocalPlayer.GetCurrentLogin
- LocalPlayer.GetLoginCount

LocalPlayer.IsChatAllowed

Brief

Returns true if the local player is not allowed to chat.

Definition

```
boolean LocalPlayer.IsChatAllowed()
```

Arguments

None.

Return Values

True if the local player is not allowed to chat, or false otherwise.

Description

If the local player is not allowed to chat, they're unable to send or receive any Text Chat, or use the Voice Comms in any way.

Examples

```
if (not LocalPlayer.IsChatAllowed())
    DoNormalUpdate()
end
```

See Also

None

LocalPlayer.IsDoingAction

Brief

Returns true if the Local Person is currently performing an action.

Definition

```
boolean LocalPlayer.IsDoingAction( boolean nonInterruptable = true )
```

Arguments

nonInterruptable - If true, only return true if the current action is not interruptable (defaults to true).

Return Values

Returns true if the Local Person is currently performing an action.

Description

Returns true if the Local Person is currently performing an action. An action is a non repeating animation, such as the action "wave".

Examples

```
print( LocalPlayer.IsDoingAction() )
```

See Also

- LocalPlayer.DoAction

LocalPlayer.IsInWardrobe

Brief

Returns true if the local player is currently located in the wardrobe.

Definition

```
boolean LocalPlayer.IsInWardrobe()
```

Arguments

None.

Return Values

True if the local player is in the wardrobe, or false otherwise.

Description

Returns true if the local player is currently located in the wardrobe.

Examples

```
if (not LocalPlayer.IsInWardrobe())
    DoNormalUpdate()
end
```

See Also

None

LocalPlayer.Relocate

Brief

Relocates the local player to the specified scene.

Definition

```
LocalPlayer.Relocate( string sceneName, number instanceParam = 0, number valueParam = 0, string
spawnFrom = nil )
```

Arguments

sceneName - The name of the scene as specified in the scene list.instanceParam - Controls the creation of different instances of the same scene (Optional - defaults to 0)valueParam - Specifies a value parameter that can be read by a script when the player has relocated to the new scene (Optional - defaults to -1)spawnFrom - [Optional] Specifies the scene name that will be used to choose the spawn location in the destination scene

Return Values

true if the destination scene name was recognised and the relocation process triggered, false if not

Description

Relocates the local player to the specified scene. Note that the relocation process is asynchronous, the user will continue to be considered present in the scene and joined to a minigame if applicable until the scene transition occurs. Therefore, expect that your script will continue to execute for some time after this call until the relocation has occurred. The following fixed scene names are also available:
BasicApartment - Harbour Studio
DEFAULT_APARTMENT - The user's default apartment

Examples

```
if PlayerHasUnlockedTheDoor() then
    LocalPlayer.Relocate( 'Secret_Room' )
end
```

See Also

- LocalPlayer.SetRelocateObject

LocalPlayer.RelocateToGroupInstance

Brief

Relocates the local player to a group instance of the specified scene.

Definition

```
LocalPlayer.RelocateToGroupInstance( string sceneName, number valueParam = nil )
```

Arguments

sceneName - The name of the scene as specified in the scene list.[optional] valueParam - A number that can be queried by the destination scene.

Return Values

True if the destination scene name was recognized and the relocation process triggered, false if not.

Description

Relocates the local player to a group instance of the specified scene. Note that the relocation process is asynchronous, the user will continue to be considered present in the scene and joined to a minigame if applicable until the scene transition occurs. Therefore, expect that your script will continue to execute for some time after this call until the relocation has occurred.

Examples

```
```

See Also

- LocalPlayer.Relocate
- LocalPlayer.RelocateToUniqueInstance

LocalPlayer.RelocateToUniqueInstance

Brief

Relocates the local player a unique instance of the specified scene.

Definition

```
LocalPlayer.RelocateToUniqueInstance( number instanceID, string sceneName )
```

Arguments

instanceID - Unique ID number of the scene (1 - 65535)sceneName - The name of the scene as specified in the scene list.[optional] valueParam - A number that can be queried by the destination scene

Return Values

true if the destination scene name was recognised and the relocation process triggered, false if not

Description

Relocates the local player to the specified scene. Note that the relocation process is asynchronous, the user will continue to be considered present in the scene and joined to a minigame if applicable until the scene transition occurs. Therefore, expect that your script will continue to execute for some time after this call until the relocation has occurred. RelocateToUniqueInstance differs from the standard Relocate method with an InstanceParam in 2 ways : 1) There will only ever be one instance of a scene with a particular UID. (If it is full, users will be denied access then returned to a new instance of their current location) 2) It is handled more efficiently on Home's backend (and hence is the preferred method for creating custom scene instances)

Examples

```
if PlayerHasSelectedPrivateSession() then
    SessionUID = 500
    LocalPlayer.RelocateToUniqueInstance( SessionUID, 'Game_Room' )
end
```

See Also

- LocalPlayer.Relocate

LocalPlayer.RemoveRepertoire

Brief

Removes the specified repertoire (by label) from the player

Definition

```
Repertoire LocalPlayer.RemoveRepertoire( string label )
```

Arguments

label - the label of the repertoire to remove.

Return Values

none

Description

Examples

```
LocalPlayer.RemoveRepertoire( "level1" )
```

See Also

None

LocalPlayer.RestoreAvatar

Brief

Returns the avatar (clothing items and rig) to the last user selection made within the wardrobe.

Definition

```
LocalPlayer.RestoreAvatar()
```

Arguments

None

Return Values

None.

Description

This method can only be used if the calling Lua environment has obtained the PersonLockType.Wardrobe lock, otherwise a lua error is thrown.

Examples

```
LocalPlayer.RestoreAvatar()
```

See Also

- LocalPlayer.SetAvatarRig
- LocalPlayer.SetRigComponentObjectId
- Person.Lock
- Person.Unlock

LocalPlayer.SetAvatarRig

Brief

Set the avatar rig type (gender).

Definition

```
boolean LocalPlayer.SetAvatarRig(RigType rigType)
```

Arguments

rigType - The rig type to change to (see description).

Return Values

True if able to change the rig (avatar is fully loaded and ready), otherwise false.

Description

Set the avatar rig type (gender) to one of the following values:
RigType.Male
RigType.Female Note that this function should be used with caution, and only with the user's consent.

Examples

```
LocalPlayer.SetAvatarRig(RigType.Male)
```

See Also

- LocalPlayer.GetAvatarRig
- Person.IsLoaded

LocalPlayer.SetClothingItems

Brief

Sets the clothing items on the player in one call

Definition

```
LocalPlayer.SetClothingItems( args )
```

Arguments

args - a table that contains the command arguments, the fields recognised are:
components [table] - a table of strings, each representing a component for the avatar to equip

Return Values

none

Description

the script must have the lock PersonLockType.Wardrobe, on the LocalPlayer person, before setting clothing items, otherwise a lua error be thrown.

Examples

```
local components = {}
table.insert(components, "00000000-00000000-00000000-01010034")
table.insert(components, "00000000-00000000-00000000-01020001")
table.insert(components, "00000000-00000000-00000000-01030009")
local setting = { components = components }
LocalPlayer.SetClothingItems( setting );
```

See Also

None

LocalPlayer.SetRelocateObject

Brief

Sets the optional objectId that can be used to script the relocation process from one scene to another

Definition

```
LocalPlayer.SetRelocateObject( string objectId, number mode = 0 )
```

Arguments

objectId - The Id of the object to usemode - The scripted relocation mode. Default 0 means the relocation script is activated once the download of the
destination scene has completed.
Mode 1 means the relocate script is started without first confirming the scene download (the scene_transition
object will need to handle the scene download)

Return Values

none

Description

Examples

```
LocalPlayer.SetRelocateObject( "12345678-12345678-12345678-12345678", 0 )
```

See Also

- LocalPlayer.Relocate

LocalPlayer.SetRigComponentObjectId

Brief

Sets the rig's component to the specified object.

Definition

```
LocalPlayer.SetRigComponentObjectId( string objectId, string componentType )
```

Arguments

objectId - The object ID for the component.componentType - The type of the rig component to set. Can be one of the following:
"Hair"
"Tors"
"Hands"
"Legs"
"Feet"
*NB This param is ignored now (1.35 and later) - works out what the componentType is when the object is loaded...

Return Values

true if successful, else false. (Will fail if the player is not ready, ie has not yet been allocated its memory slot)

Description

Sets the rig's component to the specified object. From 1.55, the script must have the lock PersonLockType.Wardrobe, on the LocalPlayer person, before setting clothing items, otherwise a lua error be thrown. The call may fail if the avatar is not in a state to accept new clothing components.

Examples

```
LocalPlayer.SetRigComponentObjectId( jacketObjectId, 'Tors' )
```

See Also

- Person.Lock

Material

Material.GetParam

Brief

Get a predefined parameter from the material

Definition

```
number|nil Material.GetParam(Material material, string name)
```

Arguments

material - The material to query.name - The parameter as it was named in maya.

Return Values

The value of the given param as it was set in maya, or nil if it doesn't exist.

Description

Gets the value of the given param as it was set in maya, or nil if it doesn't exist.

Examples

```
entity = Entity.Create( );
entity:SetModel( "mymodel" );
allMaterials = entity:FindMaterials( );
for i, material in ipairs( allMaterials ) do
    value = material:GetParam( "scrollSpeed" );
    print( "Scrollspeed is " .. tostring( value ) );
end
```

See Also

- Entity.FindMaterials

Material.SetAlphaRef

Brief

Set the alpha reference value of the material

Definition

```
Material.SetAlphaRef(Material material, number value)
```

Arguments

material - The material to set.value - The desired alpha reference value. should be between 0 and 1.

Return Values

None.

Description

Sets the alpha reference value of the material, for use with the "cutout" blend mode.

Examples

```
entity = Entity.Create( );
entity:SetModel( "mymodel" );
allMaterials = entity:FindMaterials( );
material = allMaterials[ 1 ];
material:SetBlendMode( BlendMode.Cutout );
material:SetAlphaRef( 0.1 );
```

See Also

- Entity.FindMaterials
- Material.SetBlendMode

Material.SetBlendMode

Brief

Set the blending mode of the material

Definition

```
Material.SetBlendMode(Material material, number mode)
```

Arguments

material - The material to set.mode - The desired blend mode. Valid values are:
BlendMode.Opaque
BlendMode.Cutout
BlendMode.Blend => SrcAlpha*Src + (1-SrcAlpha)*Dst
BlendMode.Add => Src + Dst
BlendMode.Multiply => Src * Dst
BlendMode.AlphaAdd => SrcAlpha*Src + Dst
BlendMode.BlendPremultiplied => Src + (1-SrcAlpha)*Dst

Return Values

None.

Description

Sets the blend mode of the material.

Examples

```
entity = Entity.Create( );
entity:SetModel( "mymodel" );
allMaterials = entity:FindMaterials( );
material = allMaterials[ 1 ];
material:SetBlendMode( BlendMode.Cutout );
material:SetAlphaRef( 0.1 );
```

See Also

- Entity.FindMaterials
- Material.SetAlphaRef

Material.SetDepthPrepass

Brief

Enable/Disable a depth prepass for this material

Definition

```
Material.SetDepthPrepass(Material material, boolean enable)
```

Arguments

material - The material to set.enable - Whether to enable or disable the depth prepass

Return Values

None.

Description

Enables (or disables) a depth prepass for use with semi-transparent materials (Removes internal pieces of a mesh).

Examples

```
entity = Entity.Create( );
entity:SetPosition( Vector4.Create( 0, 2, 0 ) );
entity:SetModel( "mymodel" );
allMaterials = entity:FindMaterials( );

material = allMaterials[ 1 ];
-- enable alpha blending
material:SetBlendMode( BlendMode.Blend );
-- enable depth prepass so the internals of the model are not visible.
material:SetDepthPrepass( true );
```

See Also

- Entity.FindMaterials
- Material.SetBlendMode

Material.SetMatrix

Brief

Set a matrix constant on the material

Definition

```
Material.SetMatrix(Material material, string name, Matrix44 value)
```

Arguments

material - The material to set.name - The uniform name.value - The value to set.

Return Values

None.

Description

Sets the given matrix shader parameter to the given value

Examples

```
rotation = 0.0;
change = 0.1;
entity = Entity.Create( );
entity:SetModel( "mymodel" );
allMaterials = entity:FindMaterials( );
material = allMaterials[ 3 ];
matrix = Matrix44.Create( );
function Update( )
    matrix:SetAxisRotation( 0, 0, 1, rotation, MATRIX_REPLACE );
    material:SetMatrix( "textureMatrix", matrix );
    rotation = rotation + change;
    if rotation > 360 or rotation < 0.0 then
        change = -change;
    end
end
```

See Also

- Entity.FindMaterials

Material.SetTexture

Brief

Set a texture on the material

Definition

```
Material.SetTexture(Material material, string name, string|Resource|Texture resource)
```

Arguments

material - The material to set.name - The uniform name.resource - The name of the texture resource, or the resource itself.

Return Values

None.

Description

Sets the given texture on the material

Examples

```
entity = Entity.Create( );
entity:SetModel( "mymodel" );
allMaterials = entity:FindMaterials( );
material = allMaterials[ 3 ];
material:SetTexture( "colourMap", "newSkin" );
```

See Also

Material.SetVector

Brief

Set a vector constant on the material

Definition

```
Material.SetVector(Material material, string name, Vector4 value)
```

Arguments

material - The material to set.name - The uniform name.value - The value to set.

Return Values

None.

Description

Sets the given vector shader parameter to the given value. The available vectors are as follows:

Name Format Applies to
---
tintClr r,g,b,a All materials with tint enabled
emissiveClr r,g,b,a All materials with emissive enabled
blendParams offset,ramp Blend materials
fresnelParams power, bias, scale Glass materials
waterColour r,g,b Water materials

Examples

```
colour = 1.0;
change = 0.01;
entity = Entity.Create( );
entity:SetModel( "gun" );
allMaterials = entity:FindMaterials( "fabric" );
for i, material in ipairs( allMaterials ) do
    material:SetBlendMode( BlendMode.Blend );
end

function Update( )
    for i, material in ipairs( allMaterials ) do
        --change the output colour
        material:SetVector( "tintClr",  Vector4.Create( colour, colour, colour, 1 ) );
    end
    colour = colour + change;
    if colour > 1.5 or colour < 0.5 then
        change = -change;
    end
end
```

See Also

- Entity.FindMaterials

Matrix44

Matrix44.Add

Brief

Performs a matrix addition with source1 and source2 storing the result in destination.

Definition

```
Matrix44.Add( Matrix44 destination, Matrix44 source1, Matrix44 source2 )
```

Arguments

destination - Matrix to receive the result.source1 - The first matrix argument for the addition.source2 - The second matrix argument for the addition.

Return Values

None.

Description

Performs a matrix addition with source1 and source2 storing the result in destination. Equivalent to destination = source1 + source2, but the operator syntax will allocate the destination automatically.

Examples

```
src1 = Matrix44.Create()
src2 = Matrix44.Create()
dst  = Matrix44.Create()
Matrix44.Add( dst, src1, src2 )
```

See Also

- Matrix44.Subtract

Matrix44.Copy

Brief

Copies the values of source and stores them in the destination Matrix44.

Definition

```
Matrix44.Copy( Matrix44 destination, Matrix44 source )
```

Arguments

destination - The matrix to receive the copy.source - The matrix to be copied.

Return Values

None

Description

Copies the values of source and stores them in the destination Matrix44. This is used to copy the values as opposed to pointing to the same matrix as an assignment would do. Use copy as mat2 = mat1 does not copy values but instead makes mat2 point to mat1.

Examples

```
mat1 = Matrix44.Create()
mat2 = Matrix44.Create()
Matrix44.Copy( mat2, mat1 )
```

See Also

None

Matrix44.Create

Brief

Creates a new Matrix44 object with the specified values.

Definition

```
Matrix44 Matrix44.Create( Vector4 x = [1,0,0,0], Vector4 y = [0,1,0,0], Vector4 z = [0,0,1,0], Vector4
w = [0,0,0,1] )
Matrix44 Matrix44.Create( Matrix44 matrix )
Matrix44 Matrix44.Create( Quaternion rotation, Vector4 translation = [0,0,0] )
```

Arguments

x - The first row of the matrix.y - The second row of the matrix.z - The third row of the matrix.w - The fourth row of the matrix.matrix - The matrix to copy.rotation - The quaternion representing the rotation part of the transform.translation - The vector representing the translation part of the transform.

Return Values

A new Matrix44 instance.

Description

Creates a new Matrix44 object with the specified values. With no parameters specified, it is set to the identity matrix.

Examples

```
mat1 = Matrix44.Create()
vec = Vector4.Create( 1, 2, 3, 4 )
mat2 = Matrix44.Create( vec, vec, vec, vec )
mat3 = Matrix44.Create( mat2 )
mat4 = Matrix44.Create( Quaternion.Create() )
```

See Also

None

Matrix44.GetRotationXyz

Brief

Gets the rotation about the x, y, and z-axis in degrees.

Definition

```
Vector4 Matrix44.GetRotationXyz( Matrix44 matrix )
Matrix44.GetRotationXyz( Matrix44 matrix, Vector4 outVec )
```

Arguments

matrix - The matrix to query.outVec - The Vector4 to store the result to.

Return Values

A vector containing the x, y, z rotation angles.

Description

Gets the rotation about the x, y, and z-axis in degrees using a matrix to Euler angle formula. The rotation order is x, y, z. If outVec is specified it stores the result in there instead of returning a new Vector4.

Examples

```
matrix = Matrix44.Create()
rotation = Matrix44.GetRotationXyz( matrix )
```

See Also

- Entity.SetRotationXyz

Matrix44.GetRow

Brief

Returns the specified row of the matrix.

Definition

```
Vector4 Matrix44.GetRow( Matrix44 matrix, number row )
Matrix44.GetRow( Matrix44 matrix, number row, Vector4 outVec )
```

Arguments

matrix - The matrix to be queried.row - The row to return, ranging from 1 to 4.outVec - The Vector4 to store the result to.

Return Values

A vector representing the specified row of the matrix.

Description

Returns the specified row of the matrix. The Vector4 is a copy of the row so changing its values will not affect the matrix. If outVec is specified it stores the result in there instead of returning a new Vector4.

Examples

```
mat = Matrix44.Create()
print( Matrix44.GetRow( mat, 1 ) )
```

See Also

- Matrix44.SetRow

Matrix44.Invert

Brief

Inverts the source matrix storing it in destination.

Definition

```
Matrix44.Invert( Matrix44 destination, Matrix44 source )
```

Arguments

destination - The matrix to receive the inverted matrix. Can be the source matrix.source - The matrix to invert.

Return Values

None

Description

Inverts the source matrix storing it in destination.

Examples

```
matrix = Matrix44.Create()
Matrix44.Invert( matrix, matrix ) -- Self inverse.
```

See Also

None

Matrix44.InvertAffine

Brief

Inverts the source matrix, which is assumed to be an affine transform, storing it in destination.

Definition

```
Matrix44.InvertAffine( Matrix44 destination, Matrix44 source )
```

Arguments

destination - The matrix to receive the inverted matrix. Can be the source matrix.source - The affine transform matrix to invert.

Return Values

None.

Description

For matrices representing an affine transform (that is, containing only scale, rotation and translation), this function will be faster than the general purpose Invert function. If your transform is only composed of rotation and translation (no scale), use the InvertAffineOrthogonal function, which is faster again.

Examples

```
matrix = Matrix44.Create()
Matrix44.InvertAffine( matrix, matrix ) -- Self inverse.
```

See Also

- Matrix44.Invert
- Matrix44.InvertAffineOrthogonal

Matrix44.InvertAffineOrthogonal

Brief

Inverts the source matrix, which is assumed to be a transform containing only rotation and translation, storing it in destination.

Definition

```
Matrix44.InvertAffineOrthogonal( Matrix44 destination, Matrix44 source )
```

Arguments

destination - The matrix to receive the inverted matrix. Can be the source matrix.source - The transform matrix to invert.

Return Values

None.

Description

If your matrix contains only rotation and translation (no scale), this function will calculate the inverse using an optimized method that is faster than both the general purpose Invert function and the InvertAffine function.

Examples

```
matrix = Matrix44.Create()
Matrix44.InvertAffineOrthogonal( matrix, matrix ) -- Self inverse.
```

See Also

- Matrix44.InvertAffine
- Matrix44.Invert

Matrix44.IsEqual

Brief

Tests whether the specified matrices are equal.

Definition

```
boolean Matrix44.IsEqual( Matrix44 m1, Matrix44 m2, number tolerance = 0 )
```

Arguments

m1 - The first matrix to test for equality.m2 - The second matrix to test for equality.tolerance - The amount of error to deviate from the other matrix.

Return Values

true if it the matrices are equal and false otherwise.

Description

Returns true if it the matrices are equal and false otherwise.

Examples

```
mat1 = Matrix44.Create()
mat2 = Matrix44.Create()
print( Matrix44.IsEqual( mat1, mat2 ) )
```

See Also

None

Matrix44.IsIdentity

Brief

Tests whether the specified matrix is an identity matrix.

Definition

```
boolean Matrix44.IsIdentity( Matrix44 matrix, number tolerance = 0 )
```

Arguments

matrix - The matrix to query.tolerance - The amount of error to deviate from the identity matrix.

Return Values

true if it is an identity matrix and false otherwise.

Description

Returns true if it is an identity matrix and false otherwise.

Examples

```
matrix = Matrix44.Create()
print( Matrix44.IsIdentity( matrix ) )
```

See Also

None

Matrix44.Multiply

Brief

Performs a matrix multiply with source1 and source2 storing the result in destination.

Definition

```
Matrix44.Multiply( Matrix44 destination, Matrix44 source1, Matrix44 source2 )
Matrix44.Multiply( Matrix44 destination, number scalar, Matrix44 matrix )
Matrix44.Multiply( Matrix44 destination, Matrix44 matrix, number scalar )
```

Arguments

source1 - The first matrix argument for the multiplication.source2 - The second matrix argument for the multiplication.scalar - Scalar to multiply.matrix - Matrix to multiply.

Return Values

None

Description

Performs a matrix multiply with source1 and source2, or a scalar multiply with scalar and matrix, storing the result in destination. Equivalent to destination = source1 * source2 or destination = scalar * matrix, but the operator syntax will allocate the destination automatically.

Examples

```
src1 = Matrix44.Create()
src2 = Matrix44.Create()
dst  = Matrix44.Create()
Matrix44.Multiply( dst, src1, src2 )
Matrix44.Multiply( dst, src1, 0.5 )
Matrix44.Multiply( dst, 2, src2 )
```

See Also

None

Matrix44.Orthonormalize

Brief

Orthonormalizes the source matrix storing it in destination.

Definition

```
Matrix44.Orthonormalize( Matrix44 destination, Matrix44 source, string axisPriority )
```

Arguments

destination - The matrix to receive the orthonormalized matrix. Can be the source matrix.source - The matrix to orthonormalize.axisPriority - A string representing the axis priority of the operation, eg "yxz".

Return Values

None

Description

Orthonormalizes the source matrix storing it in destination. The first letter in the axis priority string specifies the axis whose direction will not be changed due to the orthogonalisation. The last specifies the axis whose direction will change the most due to the orthogonalisation. Note that if it is only necessary to remove scale, it is enough to normalise the three basis vectors rather than calling this function.

Examples

```
matrix = Matrix44.Create()
Matrix44.Orthonormalize( matrix, matrix ) -- Self orthonormalize.
```

See Also

None

Matrix44.SetAxisRotation

Brief

Sets up a rotation matrix around a specified axis and angle.

Definition

```
Matrix44.SetAxisRotation( Matrix44 matrix, Vector4 rotationXyz, number angle = 0, number setType =
MATRIX_POSTMULTIPLY )
Matrix44.SetAxisRotation( Matrix44 matrix, number axisX = 1, number axisY = 1, number axisZ = 1,
number angle = 0, number setType = MATRIX_POSTMULTIPLY )
```

Arguments

matrix - The matrix to be rotated.rotationXyz - A vector containing the axis to rotate about.angle - The angle in degrees to rotate around the axis.axisX - The x element of the axis.axisY - The y element of the axis.axisZ - The z element of the axis.setType - How to set the matrix. Must be either MATRIX_POSTMULTIPLY, MATRIX_PREMULTIPLY, or MATRIX_REPLACE.

Return Values

None

Description

Sets up a rotation matrix around a specified axis and angle. Depending on the set type it either post multiplies the current matrix by the rotation matrix (matrix = matrix * rotation), pre multiplies the current matrix by the rotation matrix (matrix = rotation * matrix) or simply sets the current matrix to the rotation matrix (matrix = rotation). Non normalized axes are automatically normalized.

Examples

```
matrix = Matrix44.Create()
Matrix44.SetAxisRotation( matrix, 0.5, 0.5, 0, 90, MATRIX_REPLACE )
```

See Also

None

Matrix44.SetIdentity

Brief

Sets the matrix to the identity matrix.

Definition

```
Matrix44.SetIdentity( Matrix44 matrix )
```

Arguments

matrix - The matrix to set to identity.

Return Values

None

Description

Sets the matrix to the identity matrix.

Examples

```
matrix = Matrix44.Create()
Matrix44.SetIdentity( matrix )
```

See Also

- Matrix44.IsIdentity

Matrix44.SetRotationQuat

Brief

Sets up a rotation matrix using the specified quaternion.

Definition

```
Matrix44.SetRotationQuat( Matrix44 matrix, Quaternion quat, number setType = MATRIX_POSTMULTIPLY )
```

Arguments

matrix - The matrix to be rotated.quat - The quaternion to convert.setType - How to set the matrix. Must be either MATRIX_POSTMULTIPLY, MATRIX_PREMULTIPLY, or MATRIX_REPLACE.

Return Values

None

Description

Sets up a rotation matrix using a specified quaternion. Depending on the set type it either post multiplies the current matrix by the rotation matrix (matrix = matrix * rotation), pre multiplies the current matrix by the rotation matrix (matrix = rotation * matrix) or simply sets the current matrix to the rotation matrix (matrix = rotation).

Examples

```
matrix = Matrix44.Create()
quat = Quaternion.Create()
Matrix44.SetRotationQuat( matrix, quat, MATRIX_REPLACE )
```

See Also

None

Matrix44.SetRotationXyz

Brief

Sets up a rotation matrix from x, y, and z euler angles specified in degrees.

Definition

```
Matrix44.SetRotationXyz( Matrix44 matrix, Vector4 rotationXyz, number setType = MATRIX_POSTMULTIPLY )
Matrix44.SetRotationXyz( Matrix44 matrix, number rotationX = 1, number rotationY = 1, number rotationZ
= 1, number setType = MATRIX_POSTMULTIPLY )
```

Arguments

matrix - The matrix to be rotated.rotationXyz - A vector containing the x, y and z euler angles in degrees to set the rotation matrix to.rotationX - A number containing the x euler angle in degrees to set the rotation matrix to.rotationY - A number containing the x euler angle in degrees to set the rotation matrix to.rotationZ - A number containing the x euler angle in degrees to set the rotation matrix to.setType - How to set the matrix. Must be either MATRIX_POSTMULTIPLY, MATRIX_PREMULTIPLY, or MATRIX_REPLACE.

Return Values

None

Description

Sets up a rotation matrix from x, y, and z euler angles specified in degrees. Depending on the set type it either post multiplies the current matrix by the rotation matrix (matrix = matrix * rotation), pre multiplies the current matrix by the rotation matrix (matrix = rotation * matrix) or simply sets the current matrix to the rotation matrix (matrix = rotation). The rotation order is x, y, z.

Examples

```
matrix = Matrix44.Create()
Matrix44.SetRotationXyz( matrix, 45, 90, 180, MATRIX_REPLACE )
```

See Also

None

Matrix44.SetRow

Brief

Sets the specified row of the matrix.

Definition

```
Matrix44.SetRow( Matrix44 matrix, number row, Vector4 value )
Matrix44.SetRow( Matrix44 matrix, number row, number elem1, number elem2, number elem3, number elem4 )
```

Arguments

matrix - The matrix to be updated.row - The row to set, ranging from 1 to 4.value - A 4 dimensional vector that is copied into the first row of the matrix.elem1 - Value to set column 1 of the specified row of the matrix.elem2 - Value to set column 2 of the specified row of the matrix.elem3 - Value to set column 3 of the specified row of the matrix.elem4 - Value to set column 4 of the specified row of the matrix.

Return Values

None

Description

Sets the specified row of the matrix.

Examples

```
mat = Matrix44.Create()

-- Example 1: Set via vector
vec = Vector4.Create( 1, 2, 3, 4 )
Matrix44.SetRow( mat, 3, vec )

-- Example 2: Set via numbers
Matrix44.SetRow( mat, 3, 1, 2, 3, 4 )
```

See Also

- Matrix44.GetRow

Matrix44.SetScale

Brief

Sets up a scale matrix.

Definition

```
Matrix44.SetScale( Matrix44 matrix, Vector4 scaleXyzw, number setType = MATRIX_POSTMULTIPLY )
Matrix44.SetScale( Matrix44 matrix, number scaleX = 1, number scaleY = 1, number scaleZ = 1, number
scaleW = 1, number setType = MATRIX_POSTMULTIPLY )
```

Arguments

matrix - The matrix to be scaled.scaleXyzw - A vector containing the x, y, z and w scale to set the matrix to.scaleX - A number containing the x scale to set the matrix to.scaleY - A number containing the y scale to set the matrix to.scaleZ - A number containing the z scale to set the matrix to.scaleW - A number containing the w scale to set the matrix to.setType - How to set the matrix. Must be either MATRIX_POSTMULTIPLY, MATRIX_PREMULTIPLY, or MATRIX_REPLACE.

Return Values

None

Description

Sets up a scale matrix and depending on the set type it either post multiplies the current matrix by the scale matrix (matrix = matrix * scale), pre multiplies the current matrix by the scale matrix (matrix = scale * matrix) or simply sets the current matrix to the scale matrix (matrix = scale).

Examples

```
matrix = Matrix44.Create()
Matrix44.SetScale( matrix, 4, 3, 2, 1, MATRIX_REPLACE )
```

See Also

None

Matrix44.SetTranslation

Brief

Sets up a translation matrix

Definition

```
Matrix44.SetTranslation( Matrix44 matrix, Vector4 translationXyz, number setType = MATRIX_POSTMULTIPLY
)
Matrix44.SetTranslation( Matrix44 matrix, number translationX = 1, number translationY = 1, number
translationZ = 1, number setType = MATRIX_POSTMULTIPLY )
```

Arguments

matrix - The matrix to be translated.translationXyz - A vector containing the x, y, and z translation to set the matrix to.translationX - A number containing the x translation to set the matrix to.translationY - A number containing the y translation to set the matrix to.translationZ - A number containing the z translation to set the matrix to.setType - How to set the matrix. Must be either MATRIX_POSTMULTIPLY, MATRIX_PREMULTIPLY, or MATRIX_REPLACE.

Return Values

None

Description

Sets up a translation matrix and depending on the set type it either post multiplies the current matrix by the translation matrix (matrix = matrix * translation), pre multiplies the current matrix by the translation matrix (matrix = translation * matrix) or simply sets the current matrix to the translation matrix (matrix = translation).

Examples

```
matrix = Matrix44.Create()
Matrix44.SetTranslation( matrix, 5, 10, 5, MATRIX_REPLACE )
```

See Also

None

Matrix44.Subtract

Brief

Performs a matrix subtraction with source1 and source2 storing the result in destination.

Definition

```
Matrix44.Subtract( Matrix44 destination, Matrix44 source1, Matrix44 source2 )
```

Arguments

destination - Matrix to receive the result.source1 - The first matrix argument for the subtraction.source2 - The second matrix argument for the subtraction.

Return Values

None.

Description

Performs a matrix subtraction with source1 and source2 storing the result in destination. Equivalent to destination = source1 - source2, but the operator syntax will allocate the destination automatically.

Examples

```
src1 = Matrix44.Create()
src2 = Matrix44.Create()
dst  = Matrix44.Create()
Matrix44.Subtract( dst, src1, src2 )
```

See Also

- Matrix44.Add

Matrix44.Transpose

Brief

Transposes the source matrix storing it in destination.

Definition

```
Matrix44.Transpose( Matrix44 destination, Matrix44 source )
```

Arguments

destination - The matrix to receive the transposed matrix. Can be the source matrix.source - The matrix to transpose.

Return Values

None

Description

Transposes the source matrix storing it in destination.

Examples

```
matrix = Matrix44.Create()
Matrix44.Transpose( matrix, matrix ) -- Self transpose.
```

See Also

None

MediaLibrary

MediaLibrary.ExportImage

Brief

Moves the image file to the Cross Media Bar (XMB).

Definition

```
boolean MediaLibrary.ExportImage( string filename, string caption, function callback = nil )
boolean MediaLibrary.ExportImage( Screenshot shot, string caption, function callback = nil )
```

Arguments

filename - The filename of the image file.shot - A screenshot that has been successfully capturedcaption - The caption of the image file to be displayed in the Cross Media Bar (XMB).callback - Function to call with the export results.

Return Values

True if it started exporting and false if it is busy.

Description

Moves the image file to the Cross Media Bar (XMB). The file is moved from its current location. The operation is aysnchronous. If the callback is supplied, it will be called and passed the result of the export (boolean, true for success) and if unsuccessful, an error code.

Examples

```
if screenshot:Succeeded() == true then
    MediaLibrary.ExportImage( screenshot, "Home Screenshot", ExportCallback )
end
...
function ExportCallback(success, errorCode)
    if (success) then
        print("Picture exported!")
    else
        print("Export error:", errorCode)
    end
end
```

See Also

- MediaLibrary.ExportVideo
- Screenshot.Create

MediaLibrary.ExportVideo

Brief

Moves the video file to the Cross Media Bar (XMB).

Definition

```
boolean MediaLibrary.ExportVideo( FileHandle file, string caption, function callback = nil )
```

Arguments

file - A handle to the video file, obtained from VideoRecorder.GetOutputFile.caption - The caption of the video file to be displayed in the Cross Media Bar (XMB).

Return Values

true if it started exporting and false if it is busy.

Description

Moves the video file to the Cross Media Bar (XMB). The file is moved from its current location. The operation is aysnchronous. Note that the exporter will not handle filenames with spaces, be aware of this when recording the video.

Examples

```
local file = VideoRecorder.GetOutputFile()
MediaLibrary.ExportVideo( file, "Home Video" )
```

See Also

- MediaLibrary.ExportImage
- VideoRecorder.GetOutputFile

MediaLibrary.GetSearchResults

Brief

Gets the current search results once the search has finished.

Definition

```
table[] MediaLibrary.GetSearchResults()
```

Arguments

None

Return Values

Returns an array (as a table) of tables of the search results. If nil is returned there are no results.

Description

Gets the current search results once the search has finished. The format of the results is a table array of tables. Each table has fields included through MediaLibrary.SetResultsToInclude.

Examples

```
local results = MediaLibrary.GetSearchResults()
if results ~= nil then
    for index, result in ipairs( results ) do
        for key, value in pairs( result ) do
            print( string.format( '%4i %s = %s', index, key, value ) )
        end
    end
end

local results = MediaLibrary.GetSearchResults()
print( 'file = ' .. results[ 1 ].file )
```

See Also

- MediaLibrary.IsSearchFinished
- MediaLibrary.SetResultsToInclude

MediaLibrary.IsExporting

Brief

Checks to see if the export is in process.

Definition

```
boolean MediaLibrary.IsExporting()
```

Arguments

None

Return Values

true if it is exporting and false otherwise.

Description

Checks to see if the export is in process.

Examples

```
while MediaLibrary.IsExporting() == true do
    coroutine.yield()
end
```

See Also

None

MediaLibrary.IsSearchFinished

Brief

Checks to see if the search has finished.

Definition

```
boolean MediaLibrary.IsSearchFinished()
```

Arguments

None

Return Values

true if the search has finished and false otherwise.

Description

Checks to see if the search has finished.

Examples

```
while MediaLibrary.IsSearchFinished() == false do
    coroutine.yield()
end
```

See Also

None

MediaLibrary.IsSearching

Brief

Checks to see if the search is in process.

Definition

```
boolean MediaLibrary.IsSearching()
```

Arguments

None

Return Values

true if it is searching and false otherwise.

Description

Checks to see if the search is in process.

Examples

```
while MediaLibrary.IsSearching() == true do
    coroutine.yield()
end
```

See Also

None

MediaLibrary.Search

Brief

Starts the specified search.

Definition

```
MediaLibrary.Search( string positionType, string mediaType, number maxCount = 1 )
```

Arguments

positionType - The position to search from. Valid values are 'first', 'last', 'prev', and 'next'.mediaType - The type of media to search. Valid values are 'video', 'music', and 'photo'.maxCount - The maximum count of results to return at once. Defaults to 1.

Return Values

None

Description

Starts the specified search. You can only do one search at a time and searching is asynchronous. To get the search results use MediaLibrary.IsSearchFinished() and MediaLibrary.GetSearchResults(). The media library remembers the last position of your search for each media type so the typical way to use this would be a search from the beginning and then subsequent searches using 'next' until there are not any more results as long as you call the searches all between the same MediaLibrary.StartSearch() and MediaLibrary.StopSearch()

Examples

```
-- Get the first 10 photos only.
MediaLibrary.StartSearch()
MediaLibrary.SetResultsToInclude( 'file' )
MediaLibrary.Search( 'first', 'photo', 10 )
while MediaLibrary.IsSearchFinished() == false do
    coroutine.yield()
end
results = MediaLibrary.GetSearchResults()
if results ~= nil then
    for index, result in ipairs( results ) do
        count = count + 1
        for key, value in pairs( result ) do
            print( string.format( '%4d %4d %s = %s', count, index, key, value ) )
        end
    end
end
MediaLibrary.StopSearch()
```

See Also

- MediaLibrary.IsSearchFinished
- MediaLibrary.GetSearchResults
- MediaLibrary.SetResultsToInclude

MediaLibrary.SetPrepareSearchFiles

Brief

If true then all media files returned in the search results will be prepared.

Definition

```
MediaLibrary.SetPrepareSearchFiles( boolean value )
```

Arguments

value - If true then media files are prepared and if false they are not prepared.

Return Values

None

Description

If true then all media files returned in the search results will be prepared so they can be used, for instance in loading.

Examples

```
MediaLibrary.SetPrepareSearchFiles( true )
```

See Also

None

MediaLibrary.SetPrepareSearchThumbnails

Brief

If true then all thumbnails of media files returned in the search results will be prepared.

Definition

```
MediaLibrary.SetPrepareSearchThumbnails( boolean value )
```

Arguments

value - If true then thumbnails of the media files are prepared and if false they are not prepared.

Return Values

None

Description

If true then all thumbnails of media files returned in the search results will be prepared so they can be used, for instance in loading.

Examples

```
MediaLibrary.SetPrepareSearchThumbnails( true )
```

See Also

None

MediaLibrary.SetResultsToInclude

Brief

Sets what to include in the search results.

Definition

```
MediaLibrary.SetResultsToInclude( string includeType, boolean value = true )
```

Arguments

includeType - The type to include. Valid values are:
'file' - string, the media's filename.
'thumbnail' - string, the thumbnail filename.
'title' - string, the title of the media.
'width' - number, the width of the image if applicable.
'height' - number, the height of the image if applicable.value - If true then the type is included and if false it is excluded.

Return Values

None

Description

Sets what to include in the search results. By default nothing is included. For each field included a field is created for each media item in the array of search results.

Examples

```
MediaLibrary.StartSearch()
MediaLibrary.SetResultsToInclude( 'file', true )
MediaLibrary.SetResultsToInclude( 'title', true )
```

See Also

None

MediaLibrary.StartSearch

Brief

Prepares the media library for searching.

Definition

```
MediaLibrary.StartSearch()
```

Arguments

None

Return Values

None

Description

Prepares the media library for searching.

Examples

```
MediaLibrary.StartSearch()
```

See Also

- MediaLibrary.StopSearch

MediaLibrary.StopSearch

Brief

Stops the media library from a searching state.

Definition

```
MediaLibrary.StopSearch()
```

Arguments

None

Return Values

None

Description

Stops the media library from a searching state and clears any resources it is using. The only reason to stop searching is when you have definitely finished all your searching so you can free up the resources it has taken.

Examples

```
MediaLibrary.StopSearch()
```

See Also

- MediaLibrary.StartSearch

MemoryContainer

MemoryContainer.Create

Brief

Creates a new memory container.

Definition

```
MemoryContainer MemoryContainer.Create( number size, MemoryType memType = MemoryType.Main, number
alignment = 1 )
MemoryContainer MemoryContainer.Create( Resource resource )
```

Arguments

size - The size in bytes of the memory container.resource - A loaded file resource or based from a file resource.memType - Type of memory to

allocate.alignment - Required alignment of the memory container start address, in bytes. Must be a power of 2.

Return Values

A new instance of a memory container.

Description

Creates a new memory container. A memory container is an allocated block of memory where other library functions can output data. For example, the BasicGenx library uses a memory container to output XML which can then be posted to a server via the HttpPostData library. If creating a memory container from a resource the memory container is read only and errors will occur if attempting to write to it. Note that only file resources are accepted; textures, soundstreams and other resource types will generate an error. The container will be allocated from the script's memory budget.

Examples

```
container = MemoryContainer.Create( 4096 )
hostContainer = MemoryContainer.Create( 8192, MemoryType.Host )
myResource = Resource.Find("myFile")
resourceContainer = MemoryContainer.Create( myResource )
```

See Also

- BasicGenx.Create
- HttpPostData.Create

MemoryContainer.DebugLoad

Brief

Loads the contents of a file into a memory container from the host PC.

Definition

```
boolean MemoryContainer.DebugLoad( MemoryContainer container, string filename )
```

Arguments

container - The instance of the memory container.filename - The name of the file to load

Return Values

True if the file loaded successfully, false otherwise.

Description

Loads the contents of a file into a memory container from the host PC.

Examples

```
mc = MemoryContainer.Create( 100 )
MemoryContainer.DebugPrint( mc, "d:/temp/mc.dat" )
```

See Also

- MemoryContainer.Create

MemoryContainer.DebugPrint

Brief

Prints the contents of the container to the tty for debugging.

Definition

```
MemoryContainer.DebugPrint( MemoryContainer container )
```

Arguments

container - The instance of the memory container.

Return Values

None.

Description

Prints the contents of the container to the tty for debugging.

Examples

```
mc = MemoryContainer.Create( 100 )
MemoryContainer.SetFloat( mc, 5, 123.12 )
MemoryContainer.DebugPrint( mc )
```

See Also

- MemoryContainer.Create

MemoryContainer.DebugSave

Brief

Saves the contents of a memory container to a file on the host PC.

Definition

```
MemoryContainer.DebugSave( MemoryContainer container, string filename )
```

Arguments

container - The instance of the memory container.filename - The name of the file to save

Return Values

None.

Description

Saves the contents of a memory container to a file on the host PC.

Examples

```
mc = MemoryContainer.Create( 100 )
MemoryContainer.DebugSave( mc, "d:/temp/mc.dat" )
```

See Also

- MemoryContainer.Create

MemoryContainer.GetBigInt

Brief

Obtains the data at the specified byte index as a BigInt value.

Definition

```
void MemoryContainer.GetBigInt( MemoryContainer container, number byteIndex, BigInt result )
```

Arguments

container - The instance of the memory container.byteIndex - The byte index.result - BigInt to store value

Return Values

None.

Description

If there is insufficient data to read the entire size of the BigInt value an error will result (for example, if there are 32 bits of data remaining but the BigInt is 64 bits in size, the function will fail).

Examples

```
mc = MemoryContainer.Create( 100 )
i = BigInt.Create(64)
MemoryContainer.GetBigInt( mc, 8, i )
```

See Also

- MemoryContainer.Create
- BigInt.Create

MemoryContainer.GetBit

Brief

Returns the value of the specified bit as a boolean.

Definition

```
boolean MemoryContainer.GetBit( MemoryContainer container, number byteIndex, number bitIndex )
```

Arguments

container - The instance of the memory container.byteIndex - The byte index.bitIndex - The bit index, starts at 0 and if it is 8 or more it moves on to the next byte(s).

Return Values

true if the bit is set and false otherwise.

Description

Returns the value of the specified bit as a boolean.

Examples

```
mc = MemoryContainer.Create( 100 )
print( MemoryContainer.GetBit( mc, 5, 0 ) )
```

See Also

- MemoryContainer.Create

MemoryContainer.GetBits

Brief

Returns the value inside the specified bit range as a number.

Definition

```
number MemoryContainer.GetBits( MemoryContainer container, number byteIndex, number bitIndex, number
bitCount )
```

Arguments

container - The instance of the memory container.byteIndex - The byte index to start from.bitIndex - The bit index, starts at 0 and if it is 8 or more it moves on to the next byte(s).bitCount - The number of bits to get. The maxium is 23 due to it mapping to the mantissa of the number.

Return Values

The number contained within the bit range.

Description

Returns the number contained within the bit range. It can cross byte boundaries.

Examples

```
mc = MemoryContainer.Create( 100 )
MemoryContainer.SetInt8( mc, 255 ) )
print( MemoryContainer.GetBits( mc, 5, 1, 3 ) ) -- Get bits 1 - 4 inclusive from byte 5
```

See Also

- MemoryContainer.Create
- MemoryContainer.SetBits
- MemoryContainer.SetBit
- MemoryContainer.GetBit

MemoryContainer.GetFloat16

Brief

Returns the value of the specified 16-bit float as a number.

Definition

```
number MemoryContainer.GetFloat16( MemoryContainer container, number byteIndex )
```

Arguments

container - The instance of the memory container.byteIndex - The byte index.

Return Values

A number converted from a 16-bit float.

Description

Returns the value of the specified float as a number.

Examples

```
mc = MemoryContainer.Create( 100 )
print( MemoryContainer.GetFloat16( mc, 2 ) )
```

See Also

- MemoryContainer.Create

MemoryContainer.GetFloat32

Brief

Returns the value of the specified float as a number.

Definition

```
number MemoryContainer.GetFloat32( MemoryContainer container, number byteIndex )
```

Arguments

container - The instance of the memory container.byteIndex - The byte index.

Return Values

A number which matches the format of a float.

Description

Returns the value of the specified float as a number.

Examples

```
mc = MemoryContainer.Create( 100 )
print( MemoryContainer.GetFloat32( mc, 8 ) )
```

See Also

- MemoryContainer.Create

MemoryContainer.GetInt8

Brief

Returns the value of the specified byte as a number.

Definition

```
number MemoryContainer.GetInt8( MemoryContainer container, number byteIndex )
```

Arguments

container - The instance of the memory container.byteIndex - The byte index.

Return Values

A number ranging from -128 to 127.

Description

Returns the value of the specified byte as a number.

Examples

```
mc = MemoryContainer.Create( 100 )
print( MemoryContainer.GetInt8( mc, 5 ) )
```

See Also

- MemoryContainer.Create

MemoryContainer.GetInt16

Brief

Returns the value of the specified word as a number.

Definition

```
number MemoryContainer.GetInt16( MemoryContainer container, number byteIndex )
```

Arguments

container - The instance of the memory container.byteIndex - The byte index.

Return Values

A number ranging from -32768 to 32767.

Description

Returns the value of the specified word as a number.

Examples

```
mc = MemoryContainer.Create( 100 )
print( MemoryContainer.GetInt16( mc, 5 ) )
```

See Also

- MemoryContainer.Create

MemoryContainer.GetSize

Brief

Returns the size of the memory container in bytes.

Definition

```
number MemoryContainer.GetSize( MemoryContainer container )
```

Arguments

container - The instance of the memory container.

Return Values

The size of the memory container in bytes.

Description

Returns the size of the memory container in bytes.

Examples

```
mc = MemoryContainer.Create( 1024 )
print( MemoryContainer.GetSize( mc ) )
```

See Also

- MemoryContainer.Create

MemoryContainer.GetString

Brief

Returns a string at the specified starting position.

Definition

```
string MemoryContainer.GetString( MemoryContainer container, number byteIndex, number maxSize = nil )
```

Arguments

container - The instance of the memory container.byteIndex - The byte index to the beginning of the string.maxSize - The maximum size in bytes of the string not including the null terminator. If not specified then it stops at the first null terminator or the end of the container.

Return Values

The string created from the memory container.

Description

Returns a string at the specified starting position until it either encounters a null terminator, reaches the specified maximum size or reaches the

end of the memory container.

Examples

```
mc = MemoryContainer.Create( 100 )
MemoryContainer.SetString( mc, 0, "hello world." )
print( MemoryContainer.GetString( mc, 0 ) )
```

See Also

- MemoryContainer.SetString

MemoryContainer.GetUInt8

Brief

Returns the unsigned value of the specified byte as a number.

Definition

```
number MemoryContainer.GetUInt8( MemoryContainer container, number byteIndex )
```

Arguments

container - The instance of the memory container.byteIndex - The byte index.

Return Values

A number ranging from 0 to 255.

Description

Returns the value of the specified byte as an unsigned number.

Examples

```
mc = MemoryContainer.Create( 100 )
print( MemoryContainer.GetUInt8( mc, 5 ) )
```

See Also

- MemoryContainer.Create

MemoryContainer.GetUInt16

Brief

Returns the value of the specified word as an unsigned number.

Definition

```
number MemoryContainer.GetUInt16( MemoryContainer container, number byteIndex )
```

Arguments

container - The instance of the memory container.byteIndex - The byte index

Return Values

An unsigned number ranging from 0 to 65535.

Description

Returns the value of the specified word as an unsigned number.

Examples

```
mc = MemoryContainer.Create( 100 )
print( MemoryContainer.GetUInt16( mc, 5 ) )
```

See Also

- MemoryContainer.Create

MemoryContainer.GetUsedSize

Brief

Returns the current number of bytes written to the container.

Definition

```
MemoryContainer.GetUsedSize( MemoryContainer container )
```

Arguments

container - The instance of the memory container.

Return Values

number - The number of bytes written to the container.

Description

Returns the number of bytes written to the container using the ReadX()/WriteX() functions. The SetX() functions have no affect on this value.

Examples

```
mc = MemoryContainer.Create( 100 )
used = MemoryContainer.GetUsedSize( mc )
```

See Also

- MemoryContainer.SetUsedSize
- MemoryContainer.Tell
- MemoryContainer.Seek

MemoryContainer.MemCopy

Brief

Copy memory from one memory container to another.

Definition

```
MemoryContainer.MemCopy(MemoryContainer dest, number destOffset, MemoryContainer src, number
srcOffset, number bytes)
```

Arguments

dest - Destination memory container.destOffset - Offset into destination memory container at which to start writing.src - Source memory container.srcOffset - Offset into source memory container at which to start reading.bytes - Number of bytes to copy.

Return Values

None.

Description

Copy memory from source memory container into destination memory container, starting at respective offsets. Note that large memory copies should be avoided as this could affect the TRC compliance of your script with respect to execution time. If a large copy is required and is not required to complete in the current frame, please see AsyncCommands.StreamCopy.

Examples

```
local myResource = Resource.Find("data")
local resData = MemoryContainer.Create(myResource)
local mem = MemoryContainer.Create(resData:GetSize())
MemoryContainer.MemCopy(mem, 0, resData, 0, resData:GetSize())
```

See Also

- MemoryContainer.MemSet

MemoryContainer.MemSet

Brief

Set a number of bytes of a memory container to a specific value.

Definition

```
MemoryContainer.MemSet(MemoryContainer mem, number offset, number value, number count = nil)
```

Arguments

mem - Memory container to modify.offset - Offset into memory container at which to start writing.value - Unsigned byte value (0 - 255) to write.count - Number of bytes to set, or nil to set from offset to end of container.

Return Values

None.

Description

Sets a number of bytes in a memory container to a specific byte value. Note that large memory sets should be avoided as this could affect the TRC compliance of your script with respect to execution time.

Examples

```
local mem = MemoryContainer.Create(1024)
mem:MemSet(0, 0xff)
```

See Also

- MemoryContainer.MemCopy

MemoryContainer.ReadBigInt

Brief

Sets the value of the specified BigInt from the memory container.

Definition

```
MemoryContainer.ReadBigInt( MemoryContainer container, BigInt bigInt )
```

Arguments

container - The instance of the memory container.bigInt - The BigInt to store the value.

Return Values

None.

Description

Sets the value of the specified BigInt from the memory container. If there is insufficient space in the memory container to read the BigInt, an error will result. For example, if the container is 4 bytes from the end and you try to set a 64-bit BigInt, this function will error.

Examples

```
i = BigInt.Create(64)
MemoryContainer.ReadBigInt( mc, i )
```

See Also

- MemoryContainer.Create
- BigInt.Create
- MemoryContainer.SetBigInt
- MemoryContainer.WriteBigInt

MemoryContainer.ReadFloat16

Brief

Returns the value of the float16 at the read position float as a number.

Definition

```
number MemoryContainer.ReadFloat16( MemoryContainer container )
```

Arguments

container - The instance of the memory container.

Return Values

The value of the next float16 in the container.

Description

Returns the value of the float16 at the read position float as a number. The read position is then advanced two bytes.

Examples

```
mc = MemoryContainer.Create( 100 )
print( MemoryContainer.ReadFloat16( mc ) )
```

See Also

- MemoryContainer.Create

MemoryContainer.ReadFloat32

Brief

Returns the value of the float32 at the read position float as a number.

Definition

```
number MemoryContainer.ReadFloat32( MemoryContainer container )
```

Arguments

container - The instance of the memory container.

Return Values

The value of the next float32 in the container.

Description

Returns the value of the float32 at the read position float as a number. The read position is then advanced four bytes.

Examples

```
mc = MemoryContainer.Create( 100 )
print( MemoryContainer.ReadFloat32( mc ) )
```

See Also

- MemoryContainer.Create

MemoryContainer.ReadInt8

Brief

Returns the value of the byte at the read position float as a number.

Definition

```
number MemoryContainer.ReadInt8( MemoryContainer container )
```

Arguments

container - The instance of the memory container.

Return Values

The value of the next byte in the container.

Description

Returns the value of the byte at the read position float as a number. The read position is then advanced one byte.

Examples

```
mc = MemoryContainer.Create( 100 )
print( MemoryContainer.ReadInt8( mc ) )
```

See Also

- MemoryContainer.Create

MemoryContainer.ReadInt16

Brief

Returns the value of the int16 at the read position float as a number.

Definition

```
number MemoryContainer.ReadInt16( MemoryContainer container )
```

Arguments

container - The instance of the memory container.

Return Values

The value of the next int16 in the container.

Description

Returns the value of the int16 at the read position float as a number. The read position is then advanced two bytes.

Examples

```
mc = MemoryContainer.Create( 100 )
print( MemoryContainer.ReadInt16( mc ) )
```

See Also

- MemoryContainer.Create

MemoryContainer.ReadString

Brief

Returns the value of the string at the read position.

Definition

```
number MemoryContainer.ReadString( MemoryContainer container, readSize = 0 )
```

Arguments

container - The instance of the memory container.

Return Values

The value of the next string in the container.

Description

Reads a string from the memory container. If readSize is not specified or is zero, then the read size size is determined by searching for the terminating 0. If readSize is set, then readSize number of bytes will always be read and the string will never excede that size.

Examples

```
mc = MemoryContainer.Create( 100 )
print( MemoryContainer.ReadString( mc ) )
```

See Also

- MemoryContainer.Create

MemoryContainer.ReadUInt8

Brief

Returns the value of the byte at the read position as an unsigned number.

Definition

```
number MemoryContainer.ReadUInt8( MemoryContainer container )
```

Arguments

container - The instance of the memory container.

Return Values

The value of the 8-bit unsigned integer in the container.

Description

Returns the value of the 8-bit unsingned integer at the read position as a number. The read position is then advanced one byte.

Examples

```
mc = MemoryContainer.Create( 100 )
print( MemoryContainer.ReadUInt8( mc ) )
```

See Also

- MemoryContainer.Create

MemoryContainer.ReadUInt16

Brief

Returns the value of the uint16 at the current read position as a number.

Definition

```
number MemoryContainer.ReadUInt16( MemoryContainer container )
```

Arguments

container - The instance of the memory container.

Return Values

The value of the next uint16 in the container.

Description

Returns the value of the uint16 at the current read position as a number. The read position is then advanced two bytes.

Examples

```
mc = MemoryContainer.Create( 100 )
print( MemoryContainer.ReadUInt16( mc ) )
```

See Also

- MemoryContainer.Create
- MemoryContainer.ReadInt16

MemoryContainer.Reset

Brief

Resets a memory container.

Definition

```
MemoryContainer.Reset( MemoryContainer container )
```

Arguments

container - The instance of the memory container.

Return Values

None.

Description

Resets a memory container so that it is empty again. It will be in the same state as when it was created.

Examples

```
MemoryContainer.Reset( container )
```

See Also

- MemoryContainer.Create

MemoryContainer.Seek

Brief

Move the read/write position within the container.

Definition

```
MemoryContainer.Seek( MemoryContainer container, number position )
```

Arguments

container - The instance of the memory container.position - The new position to seek to.

Return Values

boolean - True on success.

Description

Move the read/write position within the container. This has no affect on the used size of the memory container and you can only seek within the used area of the container.

Examples

```
mc = MemoryContainer.Create( 100 )
MemoryContainer.SetUsedSize( mc, 50 )
MemoryContainer.Seek( mc, 50 )
```

See Also

- MemoryContainer.GetUsedSize
- MemoryContainer.Tell
- MemoryContainer.Seek

MemoryContainer.SetBigInt

Brief

Stores the value of the specified BigInt into the memory container at the given byte index.

Definition

```
number MemoryContainer.SetBigInt( MemoryContainer container, number byteIndex, BigInt value )
```

Arguments

container - The instance of the memory container.byteIndex - The byte index.value - The BigInt value to set into the memory container

Return Values

Returns the next byte index after the value in the BigInt is written.

Description

If there is insufficient space in the memory container to store the BigInt, an error will result. For example, if the byte offset is 4 bytes from the end of the container and you try to set a 64-bit BigInt, this function will error.

Examples

```
mc = MemoryContainer.Create( 100 )
i = BigInt.Create(64, 329823)
MemoryContainer.SetBigInt( mc, 8, i )
```

See Also

- MemoryContainer.Create
- BigInt.Create
- MemoryContainer.WriteBigInt

MemoryContainer.SetBit

Brief

Sets the value of the specified bit.

Definition

```
MemoryContainer.SetBit( MemoryContainer container, number byteIndex, number bitIndex, boolean value )
```

Arguments

container - The instance of the memory container.byteIndex - The byte index.bitIndex - The bit index, starts at 0 and if it is 8 or more it moves on to the next byte(s).value - The value to set the bit to.

Return Values

The next bit index along, can be greater than 7.

Description

Sets the value of the specified bit. If using the return value of this function and want to round it up to the next byte then do something similar to this example:
bitArray =

Unknown macro: { true, false, true, false, false, true, false, true, false, false, true, true }

mc = MemoryContainer.Create(100)
byteIndex = 23
bitIndex = 3
for index = 1, #bitArray do
bitIndex = MemoryContainer.SetBit(mc, byteIndex, bitIndex, bitArray[index])
end
byteIndex = math.floor(byteIndex + (bitIndex + 7) / 8)
MemoryContainer.SetInt8(mc, byteIndex, 255))

Examples

```
mc = MemoryContainer.Create( 100 )
MemoryContainer.SetBit( mc, 5, 1, true ) )
```

See Also

- MemoryContainer.Create

MemoryContainer.SetBits

Brief

Sets the specified range of bits to the specified value.

Definition

```
MemoryContainer.SetBits( MemoryContainer container, number byteIndex, number bitIndex, number
bitCount, number value )
```

Arguments

container - The instance of the memory container.byteIndex - The byte index.bitIndex - The bit index, starts at 0 and if it is 8 or more it moves on to the next byte(s).bitCount - The number of bits to set. Ranges from 1 to 23.value - The value to pack into the bits.

Return Values

The next bit index along, can be greater than 7.

Description

Sets the values of the specified bits.

Examples

```
mc = MemoryContainer.Create( 100 )
MemoryContainer.SetBits( mc, 5, 1, 4, 5 ) ) -- 3 bits can store values from 0 to 7.
```

See Also

- MemoryContainer.Create
- MemoryContainer.GetBits
- MemoryContainer.SetBit
- MemoryContainer.GetBit

MemoryContainer.SetFloat16

Brief

Stores the value of the specified number starting from the byte index into a Float16.

Definition

```
MemoryContainer.SetFloat16( MemoryContainer container, number byteIndex, number|number[] value )
```

Arguments

container - The instance of the memory container.byteIndex - The byte index.value - The value to set the 16-bit float to or if a table is supplied it sets all the values in the table starting from the byte index.

Return Values

Returns the next byte index after this value is written.

Description

Compresses the value of the specified to a Float16 starting from the byte index. The Float16 does not map directly to a number in lua. It is 2 bytes in size and the bits are distributed as:
bits 0 - 9 : Mantissa
bits 10 - 14 : Exponent
bit 15 : Sign
The range of the 16-bit float is between -131008 and 131008. Values out of this range are clamped to it.

Examples

```
mc = MemoryContainer.Create( 100 )
MemoryContainer.SetFloat16( mc, 5, 123.12 )
```

See Also

- MemoryContainer.Create

MemoryContainer.SetFloat32

Brief

Sets the value of the specified Float32 starting from the byte index.

Definition

```
MemoryContainer.SetFloat32( MemoryContainer container, number byteIndex, number|number[] value )
```

Arguments

container - The instance of the memory container.byteIndex - The byte index.value - The value to set in the memory container or if a table is supplied it sets all the values in the table starting from the byte index.

Return Values

Returns the next byte index after this value is written.

Description

Sets the value of the specified Float32 starting from the byte index. The Float32 maps directly to a number in lua. It is 4 bytes in size and the bits are distributed as:
bits 0 - 22 : Mantissa
bits 23 - 30 : Exponent
bit 31 : Sign

Examples

```
mc = MemoryContainer.Create( 100 )
MemoryContainer.SetFloat32( mc, 5, 123.12 )
```

See Also

- MemoryContainer.Create

MemoryContainer.SetInt8

Brief

Sets the value of the specified byte.

Definition

```
MemoryContainer.SetInt8( MemoryContainer container, number byteIndex, number|number[] value )
```

Arguments

container - The instance of the memory container.byteIndex - The byte index.value - The value to set in the memory container or if a table is supplied it sets all the values in the table starting from the byte index.

Return Values

Returns the next byte index after this value is written.

Description

Sets the value of the specified byte.

Examples

```
mc = MemoryContainer.Create( 100 )
MemoryContainer.SetInt8( mc, 5, 123 )
```

See Also

- MemoryContainer.Create

MemoryContainer.SetInt16

Brief

Sets the value of the specified word starting from the byte index.

Definition

```
MemoryContainer.SetInt16( MemoryContainer container, number byteIndex, number|number[] value )
```

Arguments

container - The instance of the memory container.byteIndex - The byte index.value - The value to set in the memory container or if a table is supplied it sets all the values in the table starting from the byte index.

Return Values

Returns the next byte index after this value is written.

Description

Sets the value of the specified word starting from the byte index.

Examples

```
mc = MemoryContainer.Create( 100 )
MemoryContainer.SetInt16( mc, 5, 12322 )
```

See Also

- MemoryContainer.Create

MemoryContainer.SetString

Brief

Stores a string in the memory container at the specified starting position.

Definition

```
MemoryContainer.SetString( MemoryContainer container, number byteIndex, string value )
```

Arguments

container - The instance of the memory container.byteIndex - The byte index to the beginning of the string.value - The string to store.

Return Values

Returns the next byte index after the string and null terminator are written.

Description

Stores a string in the memory container at the specified starting position. The memory container needs enough avaiable space to store the string and a null terminator. If there is not enough space the memory container generates an error.

Examples

```
mc = MemoryContainer.Create( 100 )
MemoryContainer.SetString( mc, 0, "hello world." )
print( MemoryContainer.GetString( mc, 0 ) )
```

See Also

- MemoryContainer.GetString

MemoryContainer.SetUsedSize

Brief

Sets the used size in bytes in the memory container.

Definition

```
MemoryContainer.GetUsedSize( MemoryContainer container, number usedSize )
```

Arguments

container - The instance of the memory container.usedSize - The number of bytes to set the used size of the container.

Return Values

Description

Sets the used size in bytes in the memory container so you can set the used size and within it and then use the ReadX()/WriteX() functions. The SetX() functions have no affect on this value.

Examples

```
mc = MemoryContainer.Create( 100 )
MemoryContainer.SetInt8( 0, 1 )
MemoryContainer.SetUsedSize( 1 )
MemoryContainer.WriteInt8( 2 )
used = MemoryContainer.SetUsedSize( 50 )
```

See Also

- MemoryContainer.GetUsedSize
- MemoryContainer.Tell
- MemoryContainer.Seek

MemoryContainer.Tell

Brief

Returns the current read/write position within the container.

Definition

```
MemoryContainer.Tell( MemoryContainer container )
```

Arguments

container - The instance of the memory container.

Return Values

number - The current read/write position in the container.

Description

Returns the current read/write position within the container.

Examples

```
mc = MemoryContainer.Create( 100 )
pos = MemoryContainer.Tell( mc )
```

See Also

- MemoryContainer.GetUsedSize
- MemoryContainer.SetUsedSize
- MemoryContainer.Seek

MemoryContainer.WriteBigInt

Brief

Writes the value of the specified BigInt into the memory container.

Definition

```
number MemoryContainer.WriteBigInt( MemoryContainer container, BigInt value )
```

Arguments

container - The instance of the memory container.value - The BigInt value to set into the memory container

Return Values

None.

Description

If there is insufficient space in the memory container to store the BigInt, an error will result. For example, if the container is 4 bytes from the end and you try to set a 64-bit BigInt, this function will error.

Examples

```
mc = MemoryContainer.Create( 100 )
i = BigInt.Create(64, 329823)
MemoryContainer.WriteBigInt( mc, i )
```

See Also

- MemoryContainer.Create
- BigInt.Create
- MemoryContainer.SetBigInt

MemoryContainer.WriteFloat16

Brief

Writes an float16 to a memory container.

Definition

```
MemoryContainer.WriteFloat16( MemoryContainer container, number|number[] value )
```

Arguments

container - The instance of the memory container.value - The value to write to the memory container or if a table then an array of values to write.

Return Values

None.

Description

Writes a float16 to a memory container and advances the write position.

Examples

```
mc = MemoryContainer.Create( 100 )
MemoryContainer.WriteFloat16( mc, 123.4 )
```

See Also

- MemoryContainer.Create

MemoryContainer.WriteFloat32

Brief

Writes a float32 to a memory container.

Definition

```
MemoryContainer.WriteFloat32( MemoryContainer container, number|number[] value )
```

Arguments

container - The instance of the memory container.value - The value to write to the memory container or if a table then an array of values to write.

Return Values

None.

Description

Writes a float32 to a memory container and advances the write position.

Examples

```
mc = MemoryContainer.Create( 100 )
MemoryContainer.WriteFloat32( mc, 123.4 )
```

See Also

- MemoryContainer.Create

MemoryContainer.WriteInt8

Brief

Writes a byte to a memory container.

Definition

```
MemoryContainer.WriteInt8( MemoryContainer container, number|number[] value )
```

Arguments

container - The instance of the memory container.value - The value to write to the memory container or if a table then an array of values to write.

Return Values

None.

Description

Writes a byte to a memory container and advances the write position.

Examples

```
mc = MemoryContainer.Create( 100 )
MemoryContainer.WriteInt8( mc, 123 )
```

See Also

- MemoryContainer.Create

MemoryContainer.WriteInt16

Brief

Writes an int16 to a memory container.

Definition

```
MemoryContainer.WriteInt16( MemoryContainer container, number|number[] value )
```

Arguments

container - The instance of the memory container.value - The value to write to the memory container or if a table then an array of values to write.

Return Values

None.

Description

Writes an int16 to a memory container and advances the write position.

Examples

```
mc = MemoryContainer.Create( 100 )
MemoryContainer.WriteInt16( mc, 123 )
```

See Also

- MemoryContainer.Create

MemoryContainer.WriteString

Brief

Writes a string to a memory container.

Definition

```
MemoryContainer.WriteString( MemoryContainer container, string value, number writeSize = 0 )
```

Arguments

container - The instance of the memory container.value - The value to write.writeSize - The number of bytes to write to the memory container

Return Values

None.

Description

Writes a string to a memory container and advances the write position. If the writeSize is specified, then writeSize number of bytes will always be written to the container.

Examples

```
mc = MemoryContainer.Create( 100 )
MemoryContainer.WriteString( mc, "Hello World!" )
```

See Also

- MemoryContainer.Create

MiniGame

MiniGame.AddCustomExitMenuOption

Brief

Add a custom menu item to the Exit Menu for the mini-game.

Definition

```
MiniGame.AddCustomExitMenuOption(string optionText)
```

Arguments

optionText - The localized string to display. Note: This must contain the "[SQUARE]" tag to denote the SQUARE button selects this option. This will almost certainly be at the start of the text, and likely include a space after it, to separate it from the rest of the text. E.g. "[SQUARE] Settings".

Return Values

None.

Description

This enables the custom menu option in the mini-game's Exit Menu, and sets the text for that option. The text must be localized before input. Only one custom option can be added. Successive calls to this function can be used to alter the text displayed. Use the 'on_custom_exit_menu_option_selected' callback to detect the SQUARE button being pressed.

Examples

```
local customText = Object.GetMe():GetLocalizedText("myCustomOptionText")
MiniGame.AddCustomExitMenuOption(myCustomOptionText)
```

See Also

- MiniGame.RemoveCustomExitMenuOption
- MiniGame.IsAbleToChangeCustomExitMenuOption

MiniGame.CloseSession

Brief

Close the minigame network session to new player entry.

Definition

```
MiniGame.CloseSession()
```

Arguments

None.

Return Values

None.

Description

Close the current minigame network session to prevent new players from joining. While the session is closed, new players will not be able to join the game. Only the session owner may call this function.

Examples

```
if (MiniGame.IsSessionOwner() and gameIsReadyToStart) then      -- some user flag indicating the lobby
period is finished
    MiniGame.CloseSession()
end
```

See Also

- MiniGame.OpenSession
- MiniGame.IsSessionOwner
- MiniGame.IsSessionOpen

MiniGame.ConfirmReceivedWelcomeMessage

Brief

Confirms that a valid welcome message has been received.

Definition

```
MiniGame.ConfirmReceivedWelcomeMessage()
```

Arguments

None

Return Values

None.

Description

Use this function to confirm that a valid welcome message has been received. Until this is called, the local player cannot join the mini-game (if welome messages are not disabled), as it is assumed they do not know the current state of it.

Examples

```
function OnReceivedMessage()
    local message = ReceivedMessage.GetEventMessage()
    local type    = ReceivedMessage.GetType( message )

    if ( type == WELCOME_MESSAGE ) then
        gameState = ReceivedMessage.ReadInt32( message )
        MiniGame.ConfirmReceivedWelcomeMessage()
    else
        -- Handle other message types
    end
end
```

See Also

- MiniGame.DisableWelcomeMessages
- MiniGame.SetSafeToSendWelcomeMessage

MiniGame.DisableWelcomeMessages

Brief

Disables welcome messages.

Definition

```
MiniGame.DisableWelcomeMessages()
```

Arguments

None

Return Values

None.

Description

You must call this function if you do not want to use welcome messages in your mini-game. A welcome message is a full refresh of game state, typically used if an avatar joins a scene and has no prior knowledge of the game. If you don't call this function, the mini-game system will expect welcome messages to be sent and received, and therefore won't allow a player to join the game until one is received from a remote player. You are advised to call this function once when your script initializes.

Examples

```
MiniGame.DisableWelcomeMessages()
```

See Also

- MiniGame.ConfirmReceivedWelcomeMessage
- MiniGame.SetSafeToSendWelcomeMessage

MiniGame.EnableQueue

Brief

When called the mini-game uses the queuing system.

Definition

```
MiniGame.EnableQueue()
```

Arguments

None

Return Values

None.

Description

When called the mini-game uses the queuing system. The default is for it to not use it. Note that this function must only be called at initialization of the script. The queuing system is totally automated and will handle joining players into the mini-game from the queue. Note: Queues act on game-types, not individual game instances. E.g. If you had 4 pool tables, the queue would be for all 4, and would place you on the first one available. The queuing system supplies queues based on the name of the mini-game. Note there's a maximum of 16 game types per scene.

Examples

```
MiniGame.EnableQueue()
```

See Also

None

MiniGame.GetCurrentFocusPersonId

Brief

Gets the person id currently in focus.

Definition

```
string MiniGame.GetCurrentFocusPersonId()
```

Arguments

None

Return Values

The person id relevant to the event callback in which it is called.

Description

Gets the currently focussed person id in the context of certain specific callbacks as specified by the 'mini_game' component in the HDK's Object Editor. This function is only valid for the following callbacks:
on_add_player
on_remove_player
on_local_player_join
on_local_player_leave
on_send_welcome_message

Examples

```
function OnAddPlayer()
    personid = MiniGame.GetCurrentFocusPersonId()
    print( "Player added: " .. personid )
end
```

See Also

- MiniGame.GetCurrentFocusPlayerSlot

MiniGame.GetCurrentFocusPlayerSlot

Brief

Gets the player slot currently in focus.

Definition

```
number MiniGame.GetCurrentFocusPlayerSlot()
```

Arguments

None

Return Values

The player slot relevant to the event callback in which it is called.

Description

Gets the currently focussed player slot in the context of certain specific callbacks as specified by the 'mini_game' component in the HDK's Object Editor. This function is only valid for the following callbacks: on_add_player on_remove_player on_local_player_join on_local_player_leave

Examples

```
function OnRemovePlayer()
    slot = MiniGame.GetCurrentFocusPlayerSlot()
    print( "Player slot of departing player: " .. slot)
end
```

See Also

- MiniGame.GetCurrentFocusPersonId

MiniGame.GetJoinType

Brief

Get the current join type of the minigame.

Definition

```
JoinType MiniGame.GetJoinType()
```

Arguments

None

Return Values

The join type of the minigame.

Description

Get the current join type of the minigame.

Examples

```
if (MiniGame.GetJoinType() == JoinType.Auto) then
    MiniGame.SetJoinType(JoinType.Normal)
end
```

See Also

- MiniGame.SetJoinType

MiniGame.GetMaxPlayerSlots

Brief

Returns the maximum number of player slots in the mini-game.

Definition

```
number MiniGame.GetMaxPlayerSlots()
```

Arguments

None

Return Values

The maximum number of player slots in the mini-game.

Description

Returns the maximum number of player slots in the mini-game. Not all of these necessarily get filled. Imagine a roulette table with 20 seats (the player slots), where only a maximum of 8 people are allowed to play at any one time.

Examples

```
print( "The max player slots are " .. MiniGame.GetMaxPlayerSlots() )
```

See Also

- MiniGame.GetCurrentFocusPlayerSlot
- MiniGame.GetPlayerSlotUsedByPersonId

MiniGame.GetNumberOfUsedPlayerSlots

Brief

Returns the number of people currently joined to the mini-game.

Definition

```
number MiniGame.GetNumberOfUsedPlayerSlots()
```

Arguments

None

Return Values

The number of people currently joined to the mini-game.

Description

Returns the number of people currently joined to the mini-game. Each person takes one player slot.

Examples

```
print( "Player slot count = " .. MiniGame.GetNumberOfUsedPlayerSlots() )
```

See Also

- MiniGame.GetCurrentFocusPlayerSlot
- MiniGame.GetPlayerSlotUsedByPersonId

MiniGame.GetPersonIdFromPlayerSlot

Brief

Returns the person ID of the given player slot. Note: Slots start from 0 (e.g. first player slot is 0, second is 1, etc).

Definition

```
string MiniGame.GetPersonIdFromPlayerSlot( number playerSlot )
```

Arguments

playerSlot - The player slot in question.

Return Values

The person ID of the given player slot.

Description

Returns the person ID of the given player slot.

Examples

```
print( "Person id of slot 0 is " .. MiniGame.GetPersonIdFromPlayerSlot( 0 ) )
```

See Also

- MiniGame.GetCurrentFocusPlayerSlot
- MiniGame.GetPlayerSlotUsedByPersonId

MiniGame.GetPlayerSlotUsedByPersonId

Brief

Returns the player slot of the specified person id.

Definition

```
number MiniGame.GetPlayerSlotUsedByPersonId( string personId )
```

Arguments

personId - Person id of an avatar that has joined the mini-game.

Return Values

The player slot of the specified player person id.

Description

Returns the player slot of the specified person id. Returns -1 if the person does not have a player slot.

Examples

```
localPlayerId = LocalPlayer.GetPersonId()
if MiniGame.IsPersonJoined( localPlayerId ) then
    slot = MiniGame.GetPlayerSlotUsedByPersonId( localPlayerId )
    print( "I am playing the game and my slot is " .. slot )
end
```

See Also

- LocalPlayer.GetPersonId
- MiniGame.IsPersonJoined

MiniGame.GetQueueSize

Brief

Used to determine the length of this mini game's queue.

Definition

```
number MiniGame.GetQueueSize()
```

Arguments

None

Return Values

Size of the queue.

Description

Returns the current size of the queue. Note: the accuracy of this value may fluctuate slightly depending on network traffic and timings etc.

Examples

```
if MiniGame.GetQueueSize() > 10 then
    TellCurrentPlayersToHurryUp()
end
```

See Also

None

MiniGame.GetTriggerRadius

Brief

Gets the mini-game's trigger radius.

Definition

```
number MiniGame.GetTriggerRadius()
```

Arguments

None

Return Values

The mini-game's trigger radius.

Description

Gets the mini-game's trigger radius in meters. This is the radius from the game position that the player must be within in order to have the option to join.

Examples

```
p = Object.GetInitialPosition( Object.GetMe() )
r = MiniGame.GetTriggerRadius()
print( "Game located at "  .. p .. " with trigger radius " .. r  )
```

See Also

- Object.GetInitialPosition
- Object.GetMe

MiniGame.IsAbleToChangeCustomExitMenuOption

Brief

Determines if a custom menu option can currently be added to the Exit Menu.

Definition

```
MiniGame.IsAbleToChangeCustomExitMenuOption()
```

Arguments

None

Return Values

Whether a custom menu option can currently be added to the Exit Menu.

Description

Changing the custom menu option while the Exit Menu is visible is not user friendly and is therefore restricted. This function can be used to determine when 'AddCustomExitMenuOption' and 'RemoveCustomExitMenuOption' will fail.

Examples

```
MiniGame.IsAbleToChangeCustomExitMenuOption()
```

See Also

- MiniGame.AddCustomExitMenuOption
- MiniGame.RemoveCustomExitMenuOption

MiniGame.IsPersonJoined

Brief

Returns true if the specified person has joined and false otherwise.

Definition

```
boolean MiniGame.IsPersonJoined( string personId )
```

Arguments

personId - The ID of the person.

Return Values

true if the specified person has joined and false otherwise.

Description

Returns true if the specified person has joined and false otherwise.

Examples

```
print( MiniGame.IsPersonJoined( LocalPlayer.GetPersonId() ) )
```

See Also

- LocalPlayer.GetPersonId

MiniGame.IsSessionOpen

Brief

Returns true if the minigame network session is currently open.

Definition

```
boolean MiniGame.IsSessionOpen()
```

Arguments

None.

Return Values

True if the network session is open, or false if closed.

Description

Returns true if the minigame network session is currently open. An open session will accept new player entry and a closed session will not.

Examples

```
if (MiniGame.IsSessionOpen() and MiniGame.IsSessionOwner()) then
    MiniGame.CloseSession()
end
```

See Also

- MiniGame.OpenSession
- MiniGame.CloseSession
- MiniGame.IsSessionOwner

MiniGame.IsSessionOwner

Brief

Returns true if the current player is the minigame network session owner.

Definition

```
boolean MiniGame.IsSessionOwner()
```

Arguments

None.

Return Values

True if the current player is the network session owner, or false otherwise.

Description

One player is designated as the network session owner for the minigame at all times. This function can be used to identify this player.

Examples

```
if (MiniGame.IsSessionOwner()) then
    print("I own this game.")
end
```

See Also

- MiniGame.OpenSession
- MiniGame.CloseSession
- MiniGame.IsSessionOpen

MiniGame.LocalPlayerLeaveGame

Brief

Triggers the local player to leave the mini-game.

Definition

```
MiniGame.LocalPlayerLeaveGame()
```

Arguments

None

Return Values

None.

Description

Triggers the local player to leave the mini-game. This will cause the function associated with the 'on_local_player_leave' property of the 'mini_game' object component to be called.

Examples

```
function OnRemovePlayer()
    -- If one player leaves the game, force everyone else to leave too
    me = LocalPlayer.GetPersonId()
    if (MiniGame.IsPersonJoined( me ) then
        MiniGame.LocalPlayerLeaveGame()
    end
end
```

See Also

- LocalPlayer.GetPersonId
- MiniGame.IsPersonJoined

MiniGame.OpenSession

Brief

Open the minigame network session to new player entry.

Definition

```
MiniGame.OpenSession()
```

Arguments

None.

Return Values

None.

Description

Open a closed session to allow new players to join. Calling this on an already open session will have no effect. Only the session owner may call this function, otherwise a Lua error will occur.

Examples

```
if (MiniGame.IsSessionOwner() and not MiniGame.IsSessionOpen()) then
    MiniGame.OpenSession()
end
```

See Also

- MiniGame.CloseSession
- MiniGame.IsSessionOwner
- MiniGame.IsSessionOpen

MiniGame.RemoveCustomExitMenuOption

Brief

Remove the custom menu item from the Exit Menu for the mini-game (if one has been added).

Definition

```
MiniGame.RemoveCustomTargetMenuOption(string optionText)
```

Arguments

None

Return Values

None.

Description

This removes the custom menu option in the mini-game's Exit Menu, if one has previously been added.

Examples

```
MiniGame.RemoveCustomExitMenuOption()
```

See Also

- MiniGame.AddCustomExitMenuOption
- MiniGame.IsAbleToChangeCustomExitMenuOption

MiniGame.SetJoinType

Brief

Set the join type of a minigame.

Definition

```
MiniGame.SetJoinType(JoinType type)
```

Arguments

type - The join type to use (JoinType.Normal, JoinType.Auto, JoinType.Minimal).

Return Values

None.

Description

Set the join type of a minigame. If the user is within range and the join type is changed to JoinType.Auto, the user will automatically join the game.

Examples

```
MiniGame.SetJoinType(JoinType.Normal)
```

See Also

- MiniGame.GetJoinType

MiniGame.SetLocalPlayerExcludeMessage

Brief

Set the message to be displayed if the minigame prevents the local player from joining.

Definition

```
MiniGame.SetLocalPlayerExcludeMessage( string message )
```

Arguments

message - the localized string to display, or an empty string to display the default message

Return Values

None.

Description

If the minigame implements the on_can_local_player_join callback, it is able to return false to indicate that the local player may not join at this time. This may be due to the game being busy or unable to accept new players at this time (for example, if a race is in progress). This function allows the script to specify a message to be displayed to the user in this event to clarify the reason for the game not being joinable, and when it will become available again. The function can be called from anywhere in the script and the message will remain active until set to another message. Note that if queueing is enabled, this function will raise an error.

Examples

```
function OnCanLocalPlayerJoin()
    if (RaceInProgress()) then
        local msg = Object.GetMe():GetLocalizedText("RACE_IN_PROGRESS")
        MiniGame.SetLocalPlayerExcludeMessage(msg)
        return false
    end
    return true
end
```

See Also

None

MiniGame.SetSafeToSendWelcomeMessage

Brief

Use this function to allow the mini-game to send welcome messages.

Definition

```
MiniGame.SetSafeToSendWelcomeMessage( boolean value = true )
```

Arguments

value - true or false. Defaults to true.

Return Values

None.

Description

Use this function to allow the mini-game to send welcome messages (if using welcome messages in the mini-game). As an example, a Pool game will not want to send welcome messages when the balls are moving. So, you should set to true at all times other than when the balls are moving.

Examples

```
function OnUpdate()
    -- Always safe for the mini-game to send a welcome message
    MiniGame.SetSafeToSendWelcomeMessage( true )
end
```

See Also

- MiniGame.ConfirmReceivedWelcomeMessage
- MiniGame.DisableWelcomeMessages

NetPropertyBag

NetPropertyBag.Create

Brief

Creates a new NetPropertyBag.

Definition

```
NetPropertyBag NetPropertyBag.Create( string bagDefinitionName )
```

Arguments

bagDefinitionName - The name of the bag definition XML file resource.

Return Values

A new instance of a network property bag.

Description

Creates a new NetPropertyBag from an XML definition resource file.

Examples

```
bag = NetPropertyBag.Create( "player" )
```

See Also

- NetPropertyBag.GetEventBag

NetPropertyBag.DeleteFromNet

Brief

Deletes a locally owned and replicated bag from the network.

Definition

```
NetPropertyBag.DeleteFromNet( NetPropertyBag bag )
```

Arguments

bag - The instance of the property bag.

Return Values

None.

Description

Deletes a locally owned and replicated bag from the network. The bag isn't deleted from memory at this point and so can be replicated again at a later time.

Examples

```
if NetPropertyBag.IsReplicated( bag ) and NetPropertyBag.IsOwnedLocally( bag ) then
    NetPropertyBag.DeleteFromNet( bag )
end
```

See Also

- NetPropertyBag.IsReplicated

- NetPropertyBag.ReplicateToNet

NetPropertyBag.GetBagIndex

Brief

Returns the index of a property bag.

Definition

```
number NetPropertyBag.GetBagIndex( NetPropertyBag bag )
```

Arguments

bag - The instance of the property bag.

Return Values

The bag index.

Description

Returns the index of a property bag.

Please be aware that this index is only unique to the client that instantiated the bag, so each bag created on a client will have a unique value, however, it is not a globally unique value across clients.

Examples

```
local bagIndex = NetPropertyBag.GetBagIndex( bag )
```

See Also

None

NetPropertyBag.GetEventBag

Brief

Returns an instance of the bag that is the focus of the current event.

Definition

```
NetPropertyBag NetPropertyBag.GetEventBag()
```

Arguments

None

Return Values

The NetPropertyBag that is the focus of the current event.

Description

Returns an instance of the bag that is the focus of the current event. Only valid during a bag created/deleted event, as named by the 'on_bag_created' and 'on_bag_deleted' properties of the 'network' object component in the HDK's Object Editor application.

Examples

```
gNetBags = { };
function OnBagCreated( )
    local bag = NetPropertyBag.GetEventBag( );
    local bagId = bag:GetTypeId( );
    --We keep a reference to a stats and skills bag which only one person creates
    if bagId == NetPropertyBag.GetTypeIdForBag( "statistics" ) then
        gNetBags[ "gamestats" ] = bag;
    elseif bagId == NetPropertyBag.GetTypeIdForBag( "skills" ) then
        gNetBags[ "skills" ] = bag;
    end
end
```

See Also

- NetPropertyBag.Create

NetPropertyBag.GetField

Brief

Gets a value from a field in the property bag.

Definition

```
variant NetPropertyBag.GetField( NetPropertyBag bag, string fieldName, table dest = nil )
```

Arguments

bag - The instance of the property bag.fieldName - The name of the field to request.dest - An optional table parameter, if provided we fill this table instead of creating one from scratch

Return Values

The boolean/number/string/table value contained in the field.

Description

Gets a value from a field in the property bag. Any client to which the bag has been replicated can call this function. Note that array fields are supported, in this case a table will be returned containing all the values in the array.

Examples

```
local name = NetPropertyBag.GetField( bag, "name" )
print(s)
local array = NetPropertyBag.GetField( bag, "array" )
for i, v in ipairs(array) do
    print(tostring(v))
end
local mytable = {}
local array = NetPropertyBag( bag, "array", mytable )
```

See Also

- NetPropertyBag.SetField

NetPropertyBag.GetOwnerId

Brief

Returns the person id of the owner of the bag.

Definition

```
string NetPropertyBag.GetOwnerId( NetPropertyBag bag )
```

Arguments

bag - The instance of the property bag.

Return Values

The person id of the game owner.

Description

Returns the person id of the owner of the bag.

Examples

```
local bagOwnerId = NetPropertyBag.GetOwnerId( bag )
bagOwner = Person.FindInInstance( bagOwnerId )
```

See Also

- Person.FindInInstance

NetPropertyBag.GetTypeId

Brief

Returns the type id of the bag.

Definition

```
number NetPropertyBag.GetTypeId( NetPropertyBag bag )
```

Arguments

bag - The instance of the property bag.

Return Values

The bag's type id.

Description

Returns the type id of the bag.

Examples

```
bag = NetPropertyBag.Create( "Player" );
if bag:GetTypeId( ) == NetPropertyBag.GetTypeIdForBag( "Player" ) then
    print( "Both bags are of the same type" );
else
    print( "Bags have different types" );
end
```

See Also

- NetPropertyBag.GetTypeIdForBag

NetPropertyBag.GetTypeIdForBag

Brief

Returns the type id for the specified property bag.

Definition

```
number NetPropertyBag.GetTypeIdForBag( string bagDefinitionName )
```

Arguments

bagDefinitionName - The definition name.

Return Values

The bag's type id.

Description

Returns the type id for the specified property bag.

Examples

```
bag = NetPropertyBag.Create( "Player" );
if bag:GetTypeId( ) == NetPropertyBag.GetTypeIdForBag( "Player" ) then
    print( "Both bags are of the same type" );
else
    print( "Bags have different types" );
end
```

See Also

- NetPropertyBag.GetTypeId

NetPropertyBag.IsOwnedLocally

Brief

Returns the local ownership state of a bag.

Definition

```
boolean NetPropertyBag.IsOwnedLocally( NetPropertyBag bag )
```

Arguments

bag - The instance of the property bag.

Return Values

True if the bag is owned locally.

Description

Returns the local ownership state of a bag.

Examples

```
local isOwnedByMe = NetPropertyBag.IsOwnedLocally( bag )
```

See Also

None

NetPropertyBag.IsReplicated

Brief

Returns the replicated state of a bag.

Definition

```
boolean NetPropertyBag.IsReplicated( NetPropertyBag bag )
```

Arguments

bag - The instance of the property bag.

Return Values

True if the bag has been replicated to the network.

Description

Returns the replicated state of a bag.

Examples

```
local isReplicated = NetPropertyBag.IsReplicated( bag )
```

See Also

- NetPropertyBag.DeleteFromNet
- NetPropertyBag.ReplicateToNet

NetPropertyBag.ReplicateToNet

Brief

Replicates a locally owned bag to the network.

Definition

```
NetPropertyBag.ReplicateToNet( NetPropertyBag bag )
```

Arguments

bag - The instance of the property bag.

Return Values

None.

Description

Replicates a locally owned bag to the network.

Examples

```
NetPropertyBag.ReplicateToNet( bag )
```

See Also

- NetPropertyBag.DeleteFromNet
- NetPropertyBag.IsReplicated

NetPropertyBag.SetField

Brief

Sets a field in the property bag.

Definition

```
NetPropertyBag.SetField( NetPropertyBag bag, string fieldName, variant value )
```

Arguments

bag - The instance of the property bag.fieldName - The name of the field to set.value - boolean/number/string/table to store in the field.

Return Values

None.

Description

Sets a field in the property bag. The bag must be owned locally for this function to succeed. Note that array fields are supported, in this case a table should be supplied of an equal or lesser number of items as the array field size. In the case of the table having less elements than the field size, the remaining elements in the field are not updated.

Examples

```
NetPropertyBag.SetField( bag, "booleanValue", true )
NetPropertyBag.SetField( bag, "numberValue", 3.14159 )
NetPropertyBag.SetField( bag, "stringValue", "Hello World" )
NetPropertyBag.SetField( bag, "arrayValue", { 1, 2, 3, 4 } )
```

See Also

- NetPropertyBag.GetField

Object

Object.DebugCreate

Brief

Instantiates an object.

Definition

```
string Object.DebugCreate( string objectId, string type, Vector4 position=[0,0,0,1], Vector4
rotation=[0,0,0,0], Vector4 scale=[1,1,1,1] )
```

Arguments

objectId - The object ID of the object to be created.position - The initial position of the object in world space.rotation - The initial rotation of the object specified as XYZ-ordered local space axis rotations in degrees.scale - The initial scale of the object.type - Type of the object. Supported types are 'portable'.

Return Values

The instance ID of the newly created object.

Description

Instantiates an object. For debug purposes only.

Examples

```
-- Create a new object
objectId = "12345678-12345678-12345678-12345678"
instanceId = Object.DebugCreate( objectId )

-- Destroy the object
Object.DebugDestroy( instanceId )
```

See Also

- Object.DebugDestroy

Object.DebugDestroy

Brief

Destroys an object.

Definition

```
Object.DebugDestroy( string instanceId )
```

Arguments

instanceId - The instance ID of the object to destroy.

Return Values

None.

Description

Destroys an object. For debug purposes only.

Examples

```
-- Create a new object
objectId = "12345678-12345678-12345678-12345678"
instanceId = Object.DebugCreate( objectId )

-- Destroy the object
Object.DebugDestroy( instanceId )
```

See Also

- Object.DebugCreate

Object.DebugEnableDebugRendering

Brief

Enable/disable debug rendering for an object.

Definition

```
Object.DebugEnableDebugRendering( Object object, boolean setEnabled )
```

Arguments

object - The object for which to enable/disable debug rendering.

Return Values

None.

Description

Sets/unsets a flag in the object to indicate that debug rendering is/is not desired. The flag is interpretted by other systems, eg the collision system.

Examples

```
me = Object.GetMe()
Object.DebugEnableDebugRendering( me, true )
```

See Also

None

Object.DebugFind

Brief

Finds the object by the specified instance ID.

Definition

```
Object Object.DebugFind( string instanceId )
```

Arguments

instanceId - The instance ID of the object to find.

Return Values

Returns the object if it is found and nil otherwise.

Description

Finds the object by the specified instance ID.

Examples

```
obj = Object.DebugFind( "0000000000000123" )
```

See Also

None

Object.DebugGetAllInstanceIds

Brief

Returns a table of all the instance IDs in the scene.

Definition

```
string[] Object.DebugGetAllInstanceIds()
```

Arguments

None

Return Values

A table of all the instance IDs in the scene

Description

Returns a table of all the instance IDs in the scene. For debug purposes only.

Examples

```
ids = Object.DebugGetAllInstanceIds()
for i=1,#ids do
    print( ids[ i ] )
end
```

See Also

None

Object.DebugReport

Brief

Reports to the tty the current state of all the objects in the instance.

Definition

```
Object.DebugReport()
```

Arguments

None

Return Values

None.

Description

Reports to the tty the current state of all the objects in the instance. For debug purposes only.

Examples

```
Object.DebugReport()
```

See Also

None

Object.DebugRun

Brief

Runs the specified Lua string in the object's Lua environment.

Definition

```
Object.DebugRun( Object object, string luaString )
```

Arguments

object - The object whose Lua environment will execute the specified command.luaString - The Lua string to run in the object's Lua environment.

Return Values

None.

Description

Runs the specified Lua string in the object's Lua environment if it exists. For debug purposes only.

Examples

```
me = Object.GetMe()
Object.DebugRun( me, "print('hi')" )
```

See Also

None

Object.EnableTargeting

Brief

Enables the targeting system for this object if it has one. Defaults to enabled.

Definition

```
Object.EnableTargeting( Object object, boolean enable )
```

Arguments

object - The object to update.enable - The state to put the targeting in for this object.

Return Values

None.

Description

Enables the targeting system for this object if it has one. Defaults to enabled. This can only be used by the lua environment in the object. The object requires a targetable component.

Examples

```
me = Object.GetMe()
if g_isLocked == true then
    Object.EnableTargeting( me, false )
end
```

See Also

None

Object.GetCurrentListenerData

Brief

Gets the listener data sent to it.

Definition

```
string|table|boolean|number Object.GetCurrentListenerData()
```

Arguments

None

Return Values

The custom result(s) from the listener callback function.

Description

Listener data is only present whilst in the 'on_listener_received_data' callback.

Examples

```
function OnListenerReceivedData()
    data = Object.GetCurrentListenerData()
    if data.type == "letter_message" then
        for i,l in ipairs( data.letters ) do
            print( 'letter ' .. i .. ' = ' .. l )
        end
    end
end
```

See Also

- Object.SendListenerData

Object.GetInitialPosition

Brief

Returns the initial position of the object.

Definition

```
Vector4 Object.GetInitialPosition( Object object )
Object.GetInitialPosition( Object object, Vector4 outPos )
```

Arguments

object - The object to query.outPos - (out) The vector to receive the position.

Return Values

The initial position of the object.

Description

Returns the initial position of the object.

Examples

```
me = Object.GetMe()
print( "My initial position is ", Object.GetInitialPosition( me ) )
```

See Also

None

Object.GetInitialRotation

Brief

Returns the initial rotation of the object.

Definition

```
Vector4 Object.GetInitialRotation( Object object )
Object.GetInitialRotation( Object object, Vector4 outRotation )
```

Arguments

object - The object to query.outRotation - (out) The vector to receive the rotation.

Return Values

Returns the initial rotation of the object.

Description

Returns the initial rotation of the object. The vector returned is the x, y, and z angles in degrees.

Examples

```
me = Object.GetMe()
print( "My initial rotation is ", Object.GetInitialRotation( me ) )
```

See Also

None

Object.GetInitialScale

Brief

Returns the initial scale of the object.

Definition

```
Vector4 Object.GetInitialScale( Object object )
Object.GetInitialScale( Object object, Vector4 outScale )
```

Arguments

object - The object to query.outScale - (out) The vector to receive the scale.

Return Values

The initial scale of the object.

Description

Returns the initial scale of the object.

Examples

```
me = Object.GetMe()
print( "My initial scale is ", Object.GetInitialScale( me ) )
```

See Also

None

Object.GetInstanceId

Brief

Returns the object's instance ID.

Definition

```
string Object.GetInstanceId( Object object )
```

Arguments

object - The object to query.

Return Values

The object's instance ID.

Description

Returns the object's instance ID.

Examples

```
obj = Object.GetMe()
print( "My instance id is " .. Object.GetInstanceId( obj ) )
```

See Also

- Object.GetMe

Object.GetInstanceName

Brief

Returns the instance name that is set in the scene editor.

Definition

```
string Object.GetInstanceName( Object object )
```

Arguments

object - The object to query.

Return Values

Returns the instance name that is set in the scene editor.

Description

Returns the instance name that is set in the scene editor. Currently supported with embedded objects and scene minigames.

Examples

```
print( 'my instance name is ', Object.GetInstanceName( Object.GetMe() ) )
```

See Also

- Object.GetNameRef
- Object.GetObjectId
- Object.GetInstanceId

Object.GetInstanceParameter

Brief

Retrieve the value of a specified instance parameter for an embedded scene object.

Definition

```
string|number|boolean|nil Object.GetInstanceParameter(Object obj, string name)
```

Arguments

obj - the object to queryname - the name of the instance parameter (not case-sensitive)

Return Values

The value of the instance parameter if it exists, or nil if it cannot be found.

Description

Gets the value of a specified instance parameter assigned to an embedded scene object. Only objects with an Embedded component will have access to the instance parameters. The names of the instance parameters are not case sensitive, and the type of the value can be string, boolean or numeric, or nil if not found. "Enum"-typed instance parameters will be returned as strings.

Examples

```
local me = Object.GetMe()
local myGameId = me:GetInstanceParameter("GameID")
if (myGameId == nil or type(myGameId) ~= "number") then
    print("Invalid game ID parameter")
end
```

See Also

Object.GetListenerObjects

Brief

Get the list of objects registered to receive messages from this object.

Definition

```
table Object.GetListenerObjects(number startIndex = 1, number maxCount = -1)
```

Arguments

startIndex - The position to start from. Previous indices are not added to the results.maxCount - The maximum size the results array can be.

Return Values

An array of Objects. The array will be empty if no listening objects are registered.

Description

Objects can register to listen for messages from other objects. This function will retrieve a list of all currently loaded objects that have registered to receive messages from the current object. If no such objects exist an empty table will be returned.

Examples

```
local listeners = Object.GetListenerObjects()
for i, obj in ipairs(listeners) do
    obj:SendListenerData("Hello!")
end
```

See Also

- Object.SendListenerData
- Scene.GetListenerObjects

Object.GetLocalizedText

Brief

Returns the localized string for the specified reference.

Definition

```
string Object.GetLocalizedText( Object object, string reference )
```

Arguments

object - The object to query.reference - The reference text to find the localization for.

Return Values

The localized string.

Description

Returns the localized string for the specified reference from the object's localization file.

Examples

```
me = Object.GetMe()
localizedGameOver = Object.GetLocalizedText( me, "GAME_OVER_MESSAGE" )
```

See Also

- Object.GetMe
- Object.GetNameRef

Object.GetMe

Brief

Returns the object that owns the script.

Definition

```
Object Object.GetMe()
```

Arguments

None

Return Values

The object that owns the script.

Description

Returns the object that owns the script.

Examples

```
me = Object.GetMe()
```

See Also

None

Object.GetNameRef

Brief

Returns the localization reference of the object's name.

Definition

```
string Object.GetNameRef( Object object )
```

Arguments

object - The object to query.

Return Values

The localization reference of the object's name.

Description

Returns the localization reference of the object's name.

Examples

```
me = Object.GetMe()
print( "My name ref is " .. Object.GetNameRef( me ) )
```

See Also

- Object.GetLocalizedText

Object.GetObjectId

Brief

Returns the object's object ID.

Definition

```
string Object.GetObjectId( Object object )
```

Arguments

object - The object to query.

Return Values

The object ID.

Description

The object's object ID.

Examples

```
me = Object.GetMe()
print( "My object id is " .. Object.GetObjectId( me ) )
```

See Also

- Object.GetMe

Object.GetPersonId

Brief

Returns the person ID that owns this object.

Definition

```
string Object.GetPersonId( Object object )
```

Arguments

object - The object to query.

Return Values

The person ID of the avatar that owns this object, or nil if the object is not owned by a person.

Description

Returns the person ID of the avatar that owns this object. Some objects, such as those placed in a clubhouse, are not considered as owned by a person, and but rather by the scene. Such objects will return nil using this function, scripts should take care to check for this case.

Examples

```
me = Object.GetMe()
ownerId = Object.GetPersonId( me )
if ( ownerId ~= nil ) then
    name = Person.GetName( ownerId )
    print( "My owner is " .. name )
end
```

See Also

- Object.GetMe

Object.HasFocus

Brief

Indicates if the object has the user's focus.

Definition

```
boolean Object.HasFocus( Object object )
```

Arguments

object - The object.

Return Values

True if the game is currently active, or false if it has lost the user input focus for any reason.

Description

Various events may take the user focus away from the currently active object, such as menus and notifications. A script may wish to use this function to poll for this case and pause the update or perform other actions while the user's focus is elsewhere. Note that due to the nature of the various actions that may take away the user focus, it is possible for this state to alternate briefly as certain systems become active or inactive. A script may wish to wait for several frames of the same focus state before reacting to a new state. The list of events that is considered to take focus away from the object is subject to change, but is currently:

- Minigame exit menu
- Menu Screen
- OSK
- Game launching OSD
- Chat log in maximized mode
- Console open (in developer builds)
- Screen adjustment
- Controller disconnection

Examples

```
paused = false
function OnUpdate()
    paused = not Object.GetMe():HasFocus()
    ...
    if (paused) then
        DoPausedLogic()
    else
        DoNormalGameLogic()
    end
end
```

See Also

None

Object.HasLocalizedText

Brief

Returns true if a localized string exists for the specified reference.

Definition

```
bool Object.HasLocalizedText( Object object, string reference )
```

Arguments

object - The object to query.reference - The reference text to find the localization for.

Return Values

True if a localized string exists for the specified reference or false otherwise.

Description

Returns true if there exists a localized string for the specified reference in the object's localization file.

Examples

```
if ( Object.HasLocalizedText( Object.GetMe(), "TEST_MESSAGE" ) ) then
    print( Object.GetLocalizedText( Object.GetMe(), "TEST_MESSAGE" ) )
end
```

See Also

- Object.GetMe
- Object.GetLocalizedText

Object.IsOwnedLocally

Brief

Returns true if the user owns this object and false otherwise.

Definition

```
boolean Object.IsOwnedLocally( Object object )
```

Arguments

object - The object to query.

Return Values

true if the user owns this object and false otherwise.

Description

Returns true if the user owns this object and false otherwise.

Examples

```
obj = Object.GetMe()
if Object.IsOwnedLocally( obj ) then
    print( "I am owned by this user" )
else
    print( "I belong to someone else" )
end
```

See Also

- Object.GetMe

Object.IsPaused

Brief

Reports if the object's Lua environment is paused.

Definition

```
boolean Object.IsPaused( Object object )
```

Arguments

object - The object to query.

Return Values

true if the specified object's Lua environment is paused and false otherwise.

Description

Reports if the Lua environment of the specified object is paused.

Examples

```
me = Object.GetMe()
print( 'pause state = ' .. Object.IsPaused( me ) )
```

See Also

- Object.Pause
- Object.Resume

Object.Pause

Brief

Pauses the lua environment.

Definition

```
Object.Pause( Object object )
```

Arguments

object - The object to be paused.

Return Values

None.

Description

Pauses the lua environment.

Examples

```
me = Object.GetMe()
Object.Pause( me )
```

See Also

- Object.Resume
- Object.IsPaused

Object.Resume

Brief

Resumes the lua environment.

Definition

```
Object.Resume( Object object )
```

Arguments

object - The object to be resumed.

Return Values

None.

Description

Resumes the lua environment if it is paused.

Examples

```
me = Object.GetMe()
Object.Resume( me )
```

See Also

- Object.Pause
- Object.IsPaused

Object.SendListenerData

Brief

Sends data from one object to another object.

Definition

```
string|table|boolean|number Object.SendListenerData( Object object, string|table|boolean|number data )
```

Arguments

object - The object to send the data to.data - The lua string to run in the object's lua environment.

Return Values

Custom return value depending on what the listener callback returns.

Description

Sends data from one object to another object.

Examples

```
sceneObject = Scene.GetVirtualObject()
t = { letters={ "a", "b", "c" }, type="letter_message" }
results = Object.SendListenerData( sceneObject, t )
print( results )
```

See Also

Object.SetFaceActiveTarget

Brief

Set whether the avatar will face the target upon activation.

Definition

```
Object.SetFaceActiveTarget(Object object, boolean value = true)
```

Arguments

object - The object to update.val - Whether the avatar will face the target upon activation.

Return Values

None.

Description

Sets whether the avatar will face the target upon activation. By default, it will face the target. The object requires a targetable component.

Examples

```
me = Object.GetMe()
Object.SetFaceActiveTarget(me, false)
```

See Also

- Object.GetMe

Object.SetTargetableLegend

Brief

Set the text of the targeting trigger (legend) for the object.

Definition

```
Object.SetTargetableLegend( Object object, string text )
```

Arguments

object - the object to update.text - the localized text to use, or nil/empty string to use the default (the object's name)

Return Values

None.

Description

Set the text of the targeting trigger (legend) for the object. This is the initial interaction trigger for the object in the scene, which brings up the targeting menu. The text should be pre-localized. The interaction symbol will be prepended to the text automatically. The object requires a targetable component.

Examples

```
me = Object.GetMe()
Object.SetTargetableLegend( me, "Play a game of pool!" )
```

See Also

- Object.SetTargetableText

Object.SetTargetablePosition

Brief

Sets the object's targetable position if it is targetable.

Definition

```
Object.SetTargetablePosition( Object object, Vector4 position )
```

Arguments

object - The object to update.position - The new targetable position

Return Values

None.

Description

Sets the object's targetable position if it is targetable using the targetable component. The position is in world space. The object requires a targetable component.

Examples

```
me = Object.GetMe()
Object.SetTargetablePosition( me, Vector4.Create( 1, 2, 3 ) )
```

See Also

- Object.GetMe

Object.SetTargetableText

Brief

Sets the text of the interact/play option in the targeting dialog.

Definition

```
Object.SetTargetableText( Object object, string text )
```

Arguments

object - The object to update.text - The localized text to display.

Return Values

None.

Description

Sets the text of the interact/play option in the targeting dialog. The object requires a targetable component.

Examples

```
me = Object.GetMe()
Object.SetTargetableText( me, 'Switch On' )
```

See Also

- Object.SetTargetableLegend

Osd

Osd.GetRoot

Brief

Returns the root OSD object.

Definition

```
OsdObject Osd.GetRoot()
```

Arguments

None

Return Values

The root OSD object.

Description

Returns the root OSD object. NOTE: From HDK 1.3.5 the OSD libraries require the developer to explicitly load each library (including any libraries created using the OSD component in Object Editor). Objects exported with an earlier version of the HDK will continue to automatically load OSD libraries as required. Loading an OSD Lua library also loads any libraries that the OSD object type depends upon. Therefore LoadLibrary("OsdBasicAnimGraphic") also loads OsdBasicSingleGraphic and OsdObject.

Examples

```
osdRoot = Osd.GetRoot()
```

See Also

None

OsdBasicAnimGraphic

OsdBasicAnimGraphic.AddFrame

Brief

Add a frame of animation to a basic anim graphic object.

Definition

```
OsdBasicAnimGraphic.AddFrame( OsdBasicAnimGraphic graphicObject, Texture back, Texture front, boolean
isReverseColor = nil )
```

Arguments

graphicObject - The basic anim graphic object to affect.back - Background texture for animation frame.front - Foreground texture for animation frame.isReverseColor - false if background/foreground textures are to be colored with background/foreground colors respectively, true to swap colors.

Return Values

None.

Description

Add a frame of animation to a basic anim graphic object. The "isReverseColor" parameter is optional and defaults to "false".

Examples

```
OsdBasicAnimGraphic.AddFrame( graphicObject, texture1, texture2, false )
```

See Also

None

OsdBasicAnimGraphic.GetNumFrames

Brief

Get the number of frames in a basic anim graphic object's animation sequence.

Definition

```
OsdBasicAnimGraphic.GetNumFrames( OsdBasicAnimGraphic graphicObject )
```

Arguments

graphicObject - The basic anim graphic object to query.

Return Values

The number of frames of animation.

Description

Get the number of frames in a basic anim graphic object's animation sequence.

Examples

```
local numFrames = OsdBasicAnimGraphic.GetNumFrames( graphicObject )
```

See Also

- OsdBasicAnimGraphic.AddFrame

OsdBasicAnimGraphic.IsAnimating

Brief

Query whether or not the basic anim graphic object is animating.

Definition

```
OsdBasicAnimGraphic.IsAnimating( OsdBasicAnimGraphic graphicObject )
```

Arguments

graphicObject - The basic anim graphic object to query.

Return Values

true if the basic anim graphic object is animating, false otherwise.

Description

Query whether or not the basic anim graphic object is animating.

Examples

```
if OsdBasicAnimGraphic.IsAnimating( graphicObject ) then
    print("AnimGraphic is animating!")
end
```

See Also

None

OsdBasicAnimGraphic.SetAnimating

Brief

Enable/disable the animation.

Definition

```
OsdBasicAnimGraphic.SetAnimating( OsdBasicAnimGraphic graphicObject, boolean setAnimating = true )
```

Arguments

graphicObject - The basic anim graphic object to affect.setAnimating - true to cause the graphic to animate, false to stop the graphic animating. Optional, defaults to true.

Return Values

None.

Description

Enable/disable the animation.

Examples

```
OsdBasicAnimGraphic.SetAnimating( graphicObject, true )
```

See Also

- OsdBasicAnimGraphic.AddFrame

OsdBasicAnimGraphic.SetCurrentFrame

Brief

Set the currently displayed frame.

Definition

```
OsdBasicAnimGraphic.SetCurrentFrame( OsdBasicAnimGraphic graphicObject, number frameIndex )
```

Arguments

graphicObject - The basic anim graphic object to affect.frameIndex - Index of frame to set as current.

Return Values

None.

Description

Set the currently displayed frame. Indices are from 1 to numFrames.

Examples

```
OsdBasicAnimGraphic.SetCurrentFrame( graphicObject, 1 )
```

See Also

- OsdBasicAnimGraphic.AddFrame

OsdBasicAnimGraphic.SetFrameRate

Brief

Set the frame rate at which to animate a basic anim graphic object.

Definition

```
OsdBasicAnimGraphic.SetFrameRate( OsdBasicAnimGraphic graphicObject, number framesPerSecond )
```

Arguments

graphicObject - The basic anim graphic object to affect.framesPerSecond - Frame rate at which to animate.

Return Values

None.

Description

Set the frame rate at which to animate a basic anim graphic object.

Examples

```
OsdBasicAnimGraphic.SetFrameRate( graphicObject, 30.0 )
```

See Also

- OsdBasicAnimGraphic.AddFrame

OsdBasicChip

OsdBasicChip.AddLegendItem

Brief

Creates a new legend item for the chip.

Definition

```
OsdBasicChip.AddLegendItem( OsdBasicChip chipObject, string localizedText )
```

Arguments

chipObject - Object to which the function is applied.text - Localized legend text.

Return Values

None.

Description

Creates a new legend item for the chip. Legend items are added left to right at the bottom of the chip. If more than one legend item is added to the chip, then they will be spaced evenly within the legend area. If more legends are added to the chip than can be displayed simultaneously, then they are cycled over time, with each legend item fading in and out of the legend area of the chip.

Examples

```
me = Object.GetMe()

-- Localized in Object Editor as "[LEFT][RIGHT][LEFTSTICK] Retarget"
legend1 = Object.GetLocalizedText( me, "RETARGET" )
-- Localized in Object Editor as "[CROSS] Select"
legend2 = Object.GetLocalizedText( me, "SELECT" )
-- Localized in Object Editor as "[CIRCLE] Exit"
legend3 = Object.GetLocalizedText( me, "EXIT" )

-- Add the 3 legend strings to the chip
OsdBasicChip.AddLegendItem( chip, legend1 )
OsdBasicChip.AddLegendItem( chip, legend2 )
OsdBasicChip.AddLegendItem( chip, legend3 )
```

See Also

- Object.GetLocalizedText
- Object.GetMe

OsdBasicChip.CreateContentObject

Brief

Creates a new OsdObject-derived object as content of the specified chip object.

Definition

```
variant OsdBasicChip.CreateContentObject( OsdBasicChip chipObject, string contentType, string
contentName )
```

Arguments

chipObject - Object to which the function is applied.contentType - A string representing the type of the content objectcontentName - The name of the content object.

Return Values

A new OsdObject-derived object of the specified type.

Description

Creates a new OsdObject-derived object as content of the specified chip object. Note: Garbage collection will not destroy OSD objects. OSD objects are only destroyed either when the parent OSD object is destroyed, or by explicitly calling OsdObject.DestroyObject (or other similar Destroy functions).

Examples

```
contentObject = OsdBasicChip.CreateContentObject( chipObject, "OsdBasicText", "mycontent" )
```

See Also

None

OsdBasicChip.RemoveAllLegendItems

Brief

Removes all legend items.

Definition

```
OsdBasicChip.RemoveAllLegendItems( OsdBasicChip chipObject )
```

Arguments

chipObject - Object to which the function is applied.

Return Values

None.

Description

Removes all legend items.

Examples

```
OsdBasicChip.RemoveAllLegendItems( chipObject )
```

See Also

None

OsdBasicChip.SetBannerHeadline

Brief

Set the headline text of an OsdBasicChip object's banner.

Definition

```
OsdBasicChip.SetBannerHeadline( OsdBasicChip chipObject, string headline )
```

Arguments

chipObject - Object to which the function is applied.headline - Headline to display on the banner.

Return Values

None.

Description

Set the headline text of an OsdBasicChip object's banner.

Examples

```
me = Object.GetMe()

-- Localized in Object Editor as "BOWLING"
headline = Object.GetLocalizedText( me, "HEADLINE_BOWLING" )

OsdBasicChip.SetBannerHeadline( chipObject, headline )
```

See Also

- Object.GetLocalizedText
- Object.GetMe

OsdBasicChip.SetBannerIconBackTexture

Brief

Set the background icon texture of an OsdBasicChip object's banner.

Definition

```
OsdBasicChip.SetBannerIconBackTexture( OsdBasicChip chipObject, Texture textureObject )
```

Arguments

chipObject - Object to which the function is applied.textureObject - Background texture object.

Return Values

None.

Description

Set the background icon texture of an OsdBasicChip object's banner.

Examples

```
OsdBasicChip.SetBannerIconBackTexture( chipObject, textureObject )
```

See Also

None

OsdBasicChip.SetBannerIconFrontTexture

Brief

Set the foreground icon texture of an OsdBasicChip object's banner.

Definition

```
OsdBasicChip.SetBannerIconFrontTexture( OsdBasicChip chipObject, Texture textureObject )
```

Arguments

chipObject - Object to which the function is applied.textureObject - Foreground texture object.

Return Values

None.

Description

Set the foreground icon texture of an OsdBasicChip object's banner.

Examples

```
OsdBasicChip.SetBannerIconFrontTexture( chipObject, textureObject )
```

See Also

None

OsdBasicChip.SetBannerIconVisible

Brief

Set the visibility of an OsdBasicChip object's banner icon.

Definition

```
OsdBasicChip.SetBannerIconVisible( OsdBasicChip chipObject, boolean setVisible )
```

Arguments

chipObject - Object to which the function is applied.setVisible - true if you want the banner icon to be visible, false otherwise.

Return Values

None.

Description

Set the visibility of an OsdBasicChip object's banner icon. The icon is invisible by default.

Examples

```
OsdBasicChip.SetBannerIconVisible( chipObject, true )
```

See Also

None

OsdBasicChip.SetBannerSubheader

Brief

Set the subheader text of an OsdBasicChip object's banner.

Definition

```
OsdBasicChip.SetBannerSubheader( OsdBasicChip chipObject, string subheader )
```

Arguments

chipObject - Object to which the function is applied.subheader - Subheader to display on the banner.

Return Values

None.

Description

Set the subheader text of an OsdBasicChip object's banner.

Examples

```
me = Object.GetMe()

-- Localized in Object Editor as "Start bowling"
subheader = Object.GetLocalizedText( me, "SUBHEADER_BOWLING" )

OsdBasicChip.SetBannerSubheader( chipObject, subheader )
```

See Also

- Object.GetLocalizedText
- Object.GetMe

OsdBasicChip.SetBannerWantVisible

Brief

Sets the chip's banner to be visible or not.

Definition

```
OsdBasicChip.SetBannerWantVisible( OsdBasicChip chipObject, boolean wantVisible )
```

Arguments

chipObject - Object to which the function is applied.wantVisible - true if you want the banner to be visible, false otherwise.

Return Values

None.

Description

Sets the chip's banner to be visible or not.

Examples

```
OsdBasicChip.SetBannerWantVisible( chipObject, true )
```

See Also

None

OsdBasicChip.SetContentHeight

Brief

Set the height of the chip's content object.

Definition

```
OsdBasicChip.SetContentHeight( OsdBasicChip chipObject, number contentHeight )
```

Arguments

chipObject - Object to which the function is applied.contentHeight - Desired height of chip's content object.

Return Values

None.

Description

Set the height of the chip's content object.

Examples

```
OsdBasicChip.SetContentHeight( chipObject, 20 )
```

See Also

- OsdBasicChip.SetContentHeightSetFromChip
- OsdBasicChip.SetContentHeightSetFromChildren

OsdBasicChip.SetContentHeightSetFromChildren

Brief

Set the height of the chip's content object to be set by its children.

Definition

```
OsdBasicChip.SetContentHeightSetFromChildren( OsdBasicChip chipObject )
```

Arguments

chipObject - Object to which the function is applied.

Return Values

None.

Description

Set the height of the chip's content object to be set by its children (i.e. the actual chip content).

Examples

```
OsdBasicChip.SetContentHeightSetFromChildren( chipObject )
```

See Also

- OsdBasicChip.SetContentHeightSetFromChip
- OsdBasicChip.SetContentHeight

OsdBasicChip.SetContentHeightSetFromChip

Brief

Set the height of the chip's content object to be set by the chip.

Definition

```
OsdBasicChip.SetContentHeightSetFromChip( OsdBasicChip chipObject )
```

Arguments

chipObject - Object to which the function is applied.

Return Values

None.

Description

Set the height of the chip's content object to be set by the chip.

Examples

```
OsdBasicChip.SetContentHeightSetFromChip( chipObject )
```

See Also

- OsdBasicChip.SetContentHeightSetFromChildren
- OsdBasicChip.SetContentHeight

OsdBasicChip.SetContentObjectInFocus

Brief

Focuses on specified content object.

Definition

```
OsdBasicChip.SetContentObjectInFocus( OsdBasicChip chipObject, OsdObject contentObject )
```

Arguments

chipObject - Object to which the function is applied.contentObject - The content object to focus on.

Return Values

None.

Description

Focuses on specified content object. Moves all content so the specified object appears as near to center as possible.

Examples

```
OsdBasicChip.SetContentObjectInFocus( chipObject, contentObject )
```

See Also

None

OsdBasicChip.SetContentPointInFocus

Brief

Focuses on specified point in content space.

Definition

```
OsdBasicChip.SetContentPointInFocus( OsdBasicChip chipObject, enum pointOrigin, number pointOffsetX =
0.0, number pointOffsetY = 0.0 )
```

Arguments

chipObject - Object to which the function is applied.pointOrigin - An enum representing the local-(ie, content-)space origin of the point to focus on. Can be one of the following:
OsdAlign.Center
OsdAlign.Top
OsdAlign.Bottom
OsdAlign.Left
OsdAlign.Right
OsdAlign.TopLeft
OsdAlign.TopRight
OsdAlign.BottomLeft
OsdAlign.BottomRightpointOffsetX - The local (content) space x-offset of the point to focus on.pointOffsetY - The local (content) space y-offset of the point to focus on.

Return Values

None.

Description

Focuses on specified point in content space. Moves all content so the specified point appears as near to the center of the window as possible.

Examples

```
-- Focuses on a point 40 pixels to the left and 20 pixels down from the top right corner of the
content
OsdBasicChip.SetContentPointInFocus( chipObject, OsdAlign.TopRight, -40, 20 )
```

See Also

None

OsdBasicChip.SetContentWidth

Brief

Set the width of the chip's content object.

Definition

```
OsdBasicChip.SetContentWidth( OsdBasicChip chipObject, number contentWidth )
```

Arguments

chipObject - Object to which the function is applied.contentWidth - Desired width of chip's content object.

Return Values

None.

Description

Set the width of the chip's content object.

Examples

```
OsdBasicChip.SetContentWidth( chipObject, 20 )
```

See Also

- OsdBasicChip.SetContentWidthSetFromChip
- OsdBasicChip.SetContentWidthSetFromChildren

OsdBasicChip.SetContentWidthSetFromChildren

Brief

Set the width of the chip's content object to be set by its children.

Definition

```
OsdBasicChip.SetContentWidthSetFromChildren(OsdBasicChip chipObject)
```

Arguments

chipObject - Object to which the function is applied.

Return Values

None.

Description

Set the width of the chip's content object to be set by its children (i.e. the actual chip content).

Examples

```
OsdBasicChip.SetContentWidthSetFromChildren( chipObject )
```

See Also

- OsdBasicChip.SetContentWidthSetFromChip
- OsdBasicChip.SetContentWidth

OsdBasicChip.SetContentWidthSetFromChip

Brief

Set the width of the chip's content object to be set by the chip.

Definition

```
OsdBasicChip.SetContentWidthSetFromChip( OsdBasicChip chipObject )
```

Arguments

chipObject - Object to which the function is applied.

Return Values

None.

Description

Set the width of the chip's content object to be set by the chip.

Examples

```
OsdBasicChip.SetContentWidthSetFromChip( chipObject )
```

See Also

- OsdBasicChip.SetContentWidthSetFromChildren
- OsdBasicChip.SetContentWidth

OsdBasicChip.SetHeightSetFromContent

Brief

Set the height of the chip to be decided by its content.

Definition

```
OsdBasicChip.SetHeightSetFromContent( OsdBasicChip chipObject, boolean isHeightSetFromContent )
```

Arguments

chipObject - Object to which the function is applied.isHeightSetFromContent - true if you want the height of the chip to be decided by its content, false otherwise.

Return Values

None.

Description

Set the height of the chip to be decided by its content.

Examples

```
OsdBasicChip.SetSize( chipObject, true )
```

See Also

None

OsdBasicChip.SetSize

Brief

Set the width of an OsdBasicChip object to a preset size.

Definition

```
OsdBasicChip.SetSize( OsdBasicChip chipObject, string size )
```

Arguments

chipObject - Object to which the function is applied.size - A string representing the size of the chip. Can be one of the following:
"quarter"
"half"
"twothirds"
"full"

Return Values

None.

Description

Set the width of an OsdBasicChip object to a preset size.

Examples

```
OsdBasicChip.SetSize( chipObject, "half" )
```

See Also

None

OsdBasicChip.UpdateContentHeight

Brief

Updates the height of the content object.

Definition

```
OsdBasicChip.UpdateContentHeight( OsdBasicChip chipObject )
```

Arguments

chipObject - Object to which the function is applied.

Return Values

None.

Description

Updates the height of the content object. To be called after adding all content if the content object height depends on its contents.

Examples

```
OsdBasicChip.UpdateContentHeight( chipObject )
```

See Also

None

OsdBasicChip.UpdateContentWidth

Brief

Updates the width of the content object.

Definition

```
OsdBasicChip.UpdateContentWidth( OsdBasicChip chipObject )
```

Arguments

chipObject - Object to which the function is applied.

Return Values

None.

Description

Updates the width of the content object. To be called after adding all content if the content object width depends on its contents.

Examples

```
OsdBasicChip.UpdateContentWidth( chipObject )
```

See Also

None

OsdBasicLegend

OsdBasicLegend.AddItem

Brief

Add a legend item to a basic legend object.

Definition

```
OsdBasicLegend.AddItem( OsdBasicLegend legendObject, string itemText, number itemId )
```

Arguments

legendObject - The basic legend object to which to add the items.itemText - The text of the legend item.itemId - The unique id of the legend item (optional parameter).

Return Values

None.

Description

Add a legend item to a basic legend object.

Examples

```
me = Object.GetMe()

-- Localized in Object Editor as "[CROSS] Continue"
legendText1 = Object.GetLocalizedText( me, "LEGEND_CONTINUE" )
-- Localized in Object Editor as "[CIRCLE] Cancel"
legendText2 = Object.GetLocalizedText( me, "LEGEND_CANCEL" )
-- Localized in Object Editor as "[SQUARE] Help"
legendText3 = Object.GetLocalizedText( me, "LEGEND_HELP" )

OsdBasicLegend.BeginAddingItems( legendObject )
OsdBasicLegend.AddItem( legendObject, legendText1, 1 )
OsdBasicLegend.AddItem( legendObject, legendText2, 2 )
OsdBasicLegend.AddItem( legendObject, legendText3, 3 )
OsdBasicLegend.EndAddingItems( legendObject )
```

See Also

- Object.GetLocalizedText
- Object.GetMe

OsdBasicLegend.BeginAddingItems

Brief

Begin adding legend items to a basic legend object.

Definition

```
OsdBasicLegend.BeginAddingItems( OsdBasicLegend legendObject )
```

Arguments

legendObject - The basic legend object to which to begin adding items.

Return Values

None.

Description

Begin adding legend items to a basic legend object. When adding legend items to a basic legend object, calling this method optimizes the process by delaying some post-processing until EndAddingItems is called.

Examples

```
me = Object.GetMe()

-- Localized in Object Editor as "[CROSS] Continue"
legendText1 = Object.GetLocalizedText( me, "LEGEND_CONTINUE" )
-- Localized in Object Editor as "[CIRCLE] Cancel"
legendText2 = Object.GetLocalizedText( me, "LEGEND_CANCEL" )
-- Localized in Object Editor as "[SQUARE] Help"
legendText3 = Object.GetLocalizedText( me, "LEGEND_HELP" )

OsdBasicLegend.BeginAddingItems( legendObject )
OsdBasicLegend.AddItem( legendObject, legendText1, 1 )
OsdBasicLegend.AddItem( legendObject, legendText2, 2 )
OsdBasicLegend.AddItem( legendObject, legendText3, 3 )
OsdBasicLegend.EndAddingItems( legendObject )
```

See Also

- Object.GetLocalizedText
- Object.GetMe

OsdBasicLegend.EndAddingItems

Brief

End adding legend items to a basic legend object.

Definition

```
OsdBasicLegend.EndAddingItems( OsdBasicLegend legendObject )
```

Arguments

legendObject - The basic legend object to which to end adding items.

Return Values

None.

Description

End adding legend items to a basic legend object. When adding legend items to a basic legend object, calling BeginAddingItems optimizes the process by delaying some post-processing until this method is called.

Examples

```
me = Object.GetMe()

-- Localized in Object Editor as "[CROSS] Continue"
legendText1 = Object.GetLocalizedText( me, "LEGEND_CONTINUE" )
-- Localized in Object Editor as "[CIRCLE] Cancel"
legendText2 = Object.GetLocalizedText( me, "LEGEND_CANCEL" )
-- Localized in Object Editor as "[SQUARE] Help"
legendText3 = Object.GetLocalizedText( me, "LEGEND_HELP" )

OsdBasicLegend.BeginAddingItems( legendObject )
OsdBasicLegend.AddItem( legendObject, legendText1, 1 )
OsdBasicLegend.AddItem( legendObject, legendText2, 2 )
OsdBasicLegend.AddItem( legendObject, legendText3, 3 )
OsdBasicLegend.EndAddingItems( legendObject )
```

See Also

- Object.GetLocalizedText
- Object.GetMe

OsdBasicLegend.RemoveAllItems

Brief

Removes all legend items from a basic legend object.

Definition

```
OsdBasicLegend.RemoveAllItems( OsdBasicLegend legendObject )
```

Arguments

legendObject - The basic legend object from which to remove all legend items.

Return Values

None.

Description

Removes all legend items from a basic legend object.

Examples

```
OsdBasicLegend.RemoveAllItems( legendObject )
```

See Also

None

OsdBasicLegend.SetBorderWidth

Brief

Set the width of the border between the left-/right-most legend items and the left/right edges of a basic legend object.

Definition

```
OsdBasicLegend.SetBorderWidth( OsdBasicLegend legendObject, number borderWidth )
```

Arguments

legendObject - The basic legend object on which to set the border width.borderWidth - The border-width of the legend object.

Return Values

None.

Description

Set the width of the border between the left-/right-most legend items and the left/right edges of a basic legend object.

Examples

```
OsdBasicLegend.SetBorderWidth( legendObject, 50.0 )
```

See Also

None

OsdBasicLegend.SetFixedWidth

Brief

Set the width of a basic legend object.

Definition

```
OsdBasicLegend.SetFixedWidth( OsdBasicLegend legendObject, number fixedWidth )
```

Arguments

legendObject - The basic legend object on which to set the maximum width.fixedWidth - The width of the legend object.

Return Values

None.

Description

Set the width of a basic legend object. Basic legend objects can resize themselves automatically based on the legend items present. This method sets the width of the object to be fixed (i.e. switches off the automatic resizing). If the legend items present demand a larger width to be completely displayed, the displayed legend items will cycle automatically.

Examples

```
OsdBasicLegend.SetFixedWidth( legendObject, 400.0 )
```

See Also

None

OsdBasicLegend.SetItemSpacing

Brief

Set the spacing between the legend items of a basic legend object.

Definition

```
OsdBasicLegend.SetItemSpacing( OsdBasicLegend legendObject, number itemSpacing )
```

Arguments

legendObject - The basic legend object on which to set the item spacing.itemSpacing - The item-spacing of the legend object.

Return Values

None.

Description

Set the spacing between the legend items of a basic legend object.

Examples

```
OsdBasicLegend.SetItemSpacing( legendObject, 50.0 )
```

See Also

None

OsdBasicLegend.SetMaxWidth

Brief

Set the maximum width of a basic legend object.

Definition

```
OsdBasicLegend.SetMaxWidth( OsdBasicLegend legendObject, number maxWidth )
```

Arguments

legendObject - The basic legend object on which to set the maximum width.maxWidth - The maximum width of the legend object.

Return Values

None.

Description

Set the maximum width of a basic legend object. Basic legend objects can resize themselves automatically based on the legend items present. This method sets a maximum width for that resizing. If the legend items present demand a larger width to be completely displayed, the displayed legend items will cycle automatically.

Examples

```
OsdBasicLegend.SetMaxWidth( legendObject, 600.0 )
```

See Also

None

OsdBasicLegend.SetMinWidth

Brief

Set the minimum width of a basic legend object.

Definition

```
OsdBasicLegend.SetMinWidth( OsdBasicLegend legendObject, number minWidth )
```

Arguments

legendObject - The basic legend object on which to set the minimum width.minWidth - The minimum width of the legend object.legendObject - The basic legend object on which to set the minimum width.minWidth - The minimum width of the legend object.

Return Values

None.

Description

Set the minimum width of a basic legend object. Basic legend objects can resize themselves automatically based on the legend items present. This method sets a minimum width for that resizing.

Examples

```
OsdBasicLegend.SetMinWidth( legendObject, 200.0 )
```

See Also

None

OsdBasicLegend.SetText

Brief

Set the text of a basic legend object.

Definition

```
OsdBasicLegend.SetText( OsdBasicLegend legendObject, string text )
```

Arguments

legendObject - The basic legend object on which to set the text.text - The localized text to display on the legend.

Return Values

None.

Description

Set the text of a basic legend object. Basic legend objects support multiple legend items. Calling this method removes all current legend items and adds a new one with the specified text.

Examples

```
me = Object.GetMe()

-- Localized in Object Editor as "[CROSS] Continue"
legendText = Object.GetLocalizedText( me, "LEGEND_CONTINUE" )

OsdBasicLegend.SetText( legendObject, legendText )
```

See Also

- Object.GetLocalizedText
- Object.GetMe

OsdBasicMenu

OsdBasicMenu.AddItem

Brief

Add a menu item to the menu object.

Definition

```
variant OsdBasicMenu.AddItem( OsdBasicMenu self, string type, number id = nil, boolean setEnabled =
true, number beforeId = nil )
```

Arguments

self - The menu object for which to call the method.type - The OSD object type of the menu item.id - Non-negative unique (in this menu) identifier for the menu item, a negative value causes the item to be allocated an id by the system.setEnabled - true to have the menu item enabled (selectable) initially, false otherwise.beforeId - the id of the existing menu item before which to insert the new one.

Return Values

The newly added OsdObject-derived menu item.

Description

Add a menu item to the menu object. The item will be added to the bottom of the menu. AddItem should be called between calls to BeginChanges and EndChanges. If the 'beforeId' argument is negative or nil, the new item will be added to the end of the menu. Note: Garbage collection will not destroy OSD objects. OSD objects are only destroyed either when the parent OSD object is destroyed, or by explicitly calling OsdObject.DestroyObject (or other similar Destroy functions).

Examples

```
menuObject = OsdObject.CreateChildObject( parentObject, "OsdBasicMenu", "menu" )
OsdBasicMenu.BeginChanges( menuObject )
itemObject = OsdBasicMenu.AddItem( menuObject, "OsdBasicSingleGraphic", 0, true )
OsdBasicSingleGraphic.SetBackTexture( itemObject, textureObject )
itemObject = OsdBasicMenu.AddItem( menuObject, "OsdCommon9ComboLayer3", 1, true )
OsdCommon9ComboLayer3.SetWidth( itemObject, 200.0 )
OsdCommon9ComboLayer3.SetHeight( itemObject, 100.0 )
OsdBasicMenu.EndChanges( menuObject )
```

See Also

None

OsdBasicMenu.BeginChanges

Brief

Prepares the menu to receive changes, such as the addition of menu items.

Definition

```
OsdBasicMenu.BeginChanges( OsdBasicMenu self )
```

Arguments

self - The menu object for which to call the method.

Return Values

None.

Description

Prepares the menu to receive changes, such as the addition of menu items. A corresponding call to EndChanges is required on completion of the changes.

Examples

```
menuObject = OsdObject.CreateChildObject( parentObject, "OsdBasicMenu", "menu" )
OsdBasicMenu.BeginChanges( menuObject )
OsdBasicMenu.AddItem( menuObject, "OsdBasicSingleGraphic", 0, true )
OsdBasicMenu.AddItem( menuObject, "OsdCommon9ComboLayer3", 1, true )
OsdBasicMenu.EndChanges( menuObject )
```

See Also

None

OsdBasicMenu.EndChanges

Brief

Tell the menu object that there are no more changes, such as adding items.

Definition

```
OsdBasicMenu.EndChanges( OsdBasicMenu self )
```

Arguments

self - The menu object for which to call the method.

Return Values

None.

Description

Tell the menu object that there are no more changes, such as adding items.

Examples

```
menuObject = OsdObject.CreateChildObject( parentObject, "OsdBasicMenu", "menu" )
OsdBasicMenu.BeginChanges( menuObject )
OsdBasicMenu.AddItem( menuObject, "OsdBasicSingleGraphic", 0, true )
OsdBasicMenu.AddItem( menuObject, "OsdCommon9ComboLayer3", 1, true )
OsdBasicMenu.EndChanges( menuObject )
```

See Also

None

OsdBasicMenu.GetCurrentItemId

Brief

Get the id of the currently selected item.

Definition

```
OsdBasicMenu.GetCurrentItemId( OsdBasicMenu self )
```

Arguments

self - The menu object for which to call the method.

Return Values

The id of the currently selected item.

Description

Get the id of the currently selected item. Assumes there exists a currently selected item, so call OsdBasicMenu.IsCurrentItemSet first.

Examples

```
menuObject = OsdObject.CreateChildObject( parentObject, "OsdBasicMenu", "menu" )
OsdBasicMenu.BeginChanges( menuObject )
OsdBasicMenu.AddItem( menuObject, "OsdBasicSingleGraphic", 0, true )
OsdBasicMenu.AddItem( menuObject, "OsdCommon9ComboLayer3", 1, true )
OsdBasicMenu.EndChanges( menuObject )
if OsdBasicMenu.IsCurrentItemSet( menuObject ) then
    id = OsdBasicMenu.GetCurrentItemId( menuObject )
    OsdBasicMenu.SetItemEnabled( menuObject, id, false )
end
```

See Also

None

OsdBasicMenu.GetItemById

Brief

Get the menu item corresponding to the specified unique id.

Definition

```
variant OsdBasicMenu.GetItemById( OsdBasicMenu self, number id )
```

Arguments

self - The menu object for which to call the method.id - The unique id of the menu item to retrieve.

Return Values

The OsdObject-derived menu item corresponding to the specified unique id.

Description

Get the menu item corresponding to the specified unique id.

Examples

```
menuObject = OsdObject.CreateChildObject( parentObject, "OsdBasicMenu", "menu" )
OsdBasicMenu.BeginChanges( menuObject )
OsdBasicMenu.AddItem( menuObject, "OsdBasicSingleGraphic", 0, true )
OsdBasicMenu.AddItem( menuObject, "OsdCommon9ComboLayer3", 1, true )
OsdBasicMenu.EndChanges( menuObject )
if OsdBasicMenu.IsCurrentItemSet( menuObject ) then
    id = OsdBasicMenu.GetCurrentItemId( menuObject )
    item = OsdBasicMenu.GetItemById( menuObject, id )
    OsdObject.SetAlphaType( item, "half" )
end
```

See Also

None

OsdBasicMenu.IsCurrentItemSet

Brief

Query whether or not the menu has a currently selected item.

Definition

```
OsdBasicMenu.IsCurrentItemSet( OsdBasicMenu self )
```

Arguments

self - The menu object for which to call the method.

Return Values

true if the menu has a currently selected item, false otherwise.

Description

Query whether or not the menu has a currently selected item. If the menu has at least one enabled item, it should have a currently selected item.

Examples

```
menuObject = OsdObject.CreateChildObject( parentObject, "OsdBasicMenu", "menu" )
OsdBasicMenu.BeginChanges( menuObject )
OsdBasicMenu.AddItem( menuObject, "OsdBasicSingleGraphic", 0, true )
OsdBasicMenu.AddItem( menuObject, "OsdCommon9ComboLayer3", 1, true )
OsdBasicMenu.EndChanges( menuObject )
if not OsdBasicMenu.IsCurrentItemSet( menuObject ) then
    OsdBasicMenu.SetCurrentItem( menuObject, 0 )
end
```

See Also

None

OsdBasicMenu.OnInputDown

Brief

Tell the menu to react to the "down" button being pressed.

Definition

```
OsdBasicMenu.OnInputDown( OsdBasicMenu self )
```

Arguments

self - The menu object for which to call the method.

Return Values

None.

Description

Tell the menu to react to the "down" button being pressed.

Examples

```
menuObject = OsdObject.CreateChildObject( parentObject, "OsdBasicMenu", "menu" )
OsdBasicMenu.BeginChanges( menuObject )
OsdBasicMenu.AddItem( menuObject, "OsdBasicSingleGraphic", 0, true )
OsdBasicMenu.AddItem( menuObject, "OsdCommon9ComboLayer3", 1, true )
OsdBasicMenu.EndChanges( menuObject )
OsdBasicMenu.OnInputDown( menuObject )
OsdBasicMenu.OnInputUp( menuObject )
OsdBasicMenu.OnInputSelect( menuObject )
```

See Also

None

OsdBasicMenu.OnInputUp

Brief

Tell the menu to react to the "up" button being pressed.

Definition

```
OsdBasicMenu.OnInputUp( OsdBasicMenu self )
```

Arguments

self - The menu object for which to call the method.

Return Values

None.

Description

Tell the menu to react to the "up" button being pressed.

Examples

```
menuObject = OsdObject.CreateChildObject( parentObject, "OsdBasicMenu", "menu" )
OsdBasicMenu.BeginChanges( menuObject )
OsdBasicMenu.AddItem( menuObject, "OsdBasicSingleGraphic", 0, true )
OsdBasicMenu.AddItem( menuObject, "OsdCommon9ComboLayer3", 1, true )
OsdBasicMenu.EndChanges( menuObject )
OsdBasicMenu.OnInputDown( menuObject )
OsdBasicMenu.OnInputUp( menuObject )
OsdBasicMenu.OnInputSelect( menuObject )
```

See Also

None

OsdBasicMenu.Reset

Brief

Removes and destroys all menu items.

Definition

```
OsdBasicMenu.Reset( OsdBasicMenu self )
```

Arguments

self - The menu object for which to call the method.

Return Values

None.

Description

Removes and destroys all menu items.

Examples

```
menuObject = OsdObject.CreateChildObject( parentObject, "OsdBasicMenu", "menu" )
OsdBasicMenu.BeginChanges( menuObject )
OsdBasicMenu.AddItem( menuObject, "OsdBasicSingleGraphic", 0, true )
OsdBasicMenu.AddItem( menuObject, "OsdCommon9ComboLayer3", 1, true )
OsdBasicMenu.EndChanges( menuObject )
OsdBasicMenu.Reset( menuObject )
```

See Also

None

OsdBasicMenu.SetCurrentItem

Brief

Set the specified menu item to be the currently selected one.

Definition

```
OsdBasicMenu.SetCurrentItem( OsdBasicMenu self, number itemId )
```

Arguments

self - The menu object for which to call the method.itemId - The unique id of the menu item.

Return Values

None.

Description

Set the specified menu item to be the currently selected one. Mainly for initialisation; the OnInput... methods should be used to change the currently selected menu item after that.

Examples

```
menuObject = OsdObject.CreateChildObject( parentObject, "OsdBasicMenu", "menu" )
OsdBasicMenu.BeginChanges( menuObject )
OsdBasicMenu.AddItem( menuObject, "OsdBasicSingleGraphic", 0, true )
OsdBasicMenu.AddItem( menuObject, "OsdCommon9ComboLayer3", 1, true )
OsdBasicMenu.EndChanges( menuObject )
OsdBasicMenu.SetCurrentItem( menuObject, 0 )
```

See Also

None

OsdBasicMenu.SetItemEnabled

Brief

Set the specified menu item to be enabled/disabled.

Definition

```
OsdBasicMenu.SetItemEnabled( OsdBasicMenu self, number itemId, boolean setEnabled )
```

Arguments

self - The menu object for which to call the method.itemId - The unique id of the menu item.setEnabled - true if enabling the item, false otherwise.

Return Values

None.

Description

Set the specified menu item to be enabled/disabled.

Examples

```
menuObject = OsdObject.CreateChildObject( parentObject, "OsdBasicMenu", "menu" )
OsdBasicMenu.BeginChanges( menuObject )
OsdBasicMenu.AddItem( menuObject, "OsdBasicSingleGraphic", 0, true )
OsdBasicMenu.AddItem( menuObject, "OsdCommon9ComboLayer3", 1, true )
OsdBasicMenu.EndChanges( menuObject )
OsdBasicMenu.SetItemEnabled( menuObject, 1, false )
```

See Also

None

OsdBasicMenu.SetItemSpacing

Brief

Set the vertical spacing between menu items.

Definition

```
OsdBasicMenu.SetItemSpacing( OsdBasicMenu self, number itemSpacing )
```

Arguments

self - The menu object for which to call the method.itemSpacing - The spacing in pixels.

Return Values

None.

Description

Set the vertical spacing between menu items. Should be called between calls to BeginChanges and EndChanges. Can be called before or after adding menu items.

Examples

```
menuObject = OsdObject.CreateChildObject( parentObject, "OsdBasicMenu", "menu" )
OsdBasicMenu.BeginChanges( menuObject )
OsdBasicMenu.SetItemSpacing( menuObject, 40.0 )
OsdBasicMenu.AddItem( menuObject, "OsdBasicSingleGraphic", 0, true )
OsdBasicMenu.AddItem( menuObject, "OsdCommon9ComboLayer3", 1, true )
OsdBasicMenu.EndChanges( menuObject )
```

See Also

None

OsdBasicMenu.SetOnSetItemEnabledCallback

Brief

Set the lua callback function to be called when a menu item is enabled/disabled.

Definition

```
OsdBasicMenu.SetOnSetItemEnabledCallback( OsdBasicMenu self, function callback )
```

Arguments

self - The menu object for which to call the method.callback - The callback function.

Return Values

None.

Description

Set the lua callback function to be called when a menu item is enabled/disabled. The function should have two parameters; an OsdObject (the item) and a boolean (true if the item is being enabled, false otherwise).

Examples

```
function GameMenu.SetItemEnabledCallback( item, setEnabled )
    if setEnabled then
        OsdObject.SetColorType( item, "fg" )
    else
        OsdObject.SetColorType( item, "grey" )
    end
end

function GameMenu.Create( parentObject )
    menuObject = OsdObject.CreateChildObject( parentObject, "OsdBasicMenu", "menu" )
    OsdBasicMenu.SetOnSetItemEnabledCallback( menuObject, GameMenu.SetItemEnabledCallback )
    OsdBasicMenu.BeginChanges( menuObject )
    OsdBasicMenu.SetItemSpacing( menuObject, 40.0 )
    OsdBasicMenu.AddItem( menuObject, "OsdBasicSingleGraphic", 0, true )
    OsdBasicMenu.AddItem( menuObject, "OsdCommon9ComboLayer3", 1, true )
    OsdBasicMenu.EndChanges( menuObject )
end
```

See Also

None

OsdBasicMenu.SetOnSetItemSelectedCallback

Brief

Set the lua callback function to be called when a menu item is selected/unselected.

Definition

```
OsdBasicMenu.SetOnSetItemSelectedCallback( OsdBasicMenu self, function callback )
```

Arguments

self - The menu object for which to call the method.callback - The callback function.

Return Values

None.

Description

Set the lua callback function to be called when a menu item is selected/unselected. The function should have two parameters; an OsdObject (the item) and a boolean (true if the item is being selected, false otherwise).

Examples

```
function GameMenu.SetItemSelectedCallback( item, setSelected )
    if setSelected then
        OsdObject.SetAlphaType( item, "normal" )
    else
        OsdObject.SetAlphaType( item, "half" )
    end
end
function GameMenu.Create( parentObject )
    menuObject = OsdObject.CreateChildObject( parentObject, "OsdBasicMenu", "menu" )
    OsdBasicMenu.SetOnSetItemSelectedCallback( menuObject, GameMenu.SetItemSelectedCallback )
    OsdBasicMenu.BeginChanges( menuObject )
    OsdBasicMenu.SetItemSpacing( menuObject, 40.0 )
    OsdBasicMenu.AddItem( menuObject, "OsdBasicSingleGraphic", 0, true )
    OsdBasicMenu.AddItem( menuObject, "OsdCommon9ComboLayer3", 1, true )
    OsdBasicMenu.EndChanges( menuObject )
end
```

See Also

None

OsdBasicPopup

OsdBasicPopup.SetMessage

Brief

Set the message text on a basic popup object.

Definition

```
OsdBasicPopup.SetMessage( OsdBasicPopup popupObject, string messageText )
```

Arguments

popupObject - The basic popup object for which to set the title text.messageText - The localized message text.

Return Values

None.

Description

Set the message text on a basic popup object. Basic popup objects consist of title text and message text.

Examples

```
me = Object.GetMe()

-- Localized in Object Editor as "WARNING!"
title   = Object.GetLocalizedText( me, "WARNING_POPUP_TITLE" )
-- Localized in Object Editor as "Something has gone wrong!"
message = Object.GetLocalizedText( me, "WARNING_POPUP_MESSAGE" )

OsdBasicPopup.SetTitle( popup, title )
OsdBasicPopup.SetMessage( popup, message )
```

See Also

- Object.GetLocalizedText
- Object.GetMe

OsdBasicPopup.SetTitle

Brief

Set the title text on a basic popup object.

Definition

```
OsdBasicPopup.SetTitle( OsdBasicPopup popupObject, string titleText )
```

Arguments

popupObject - The basic popup object for which to set the title text.titleText - The localized title text.

Return Values

None.

Description

Set the title text on a basic popup object. Basic popup objects consist of title text and message text.

Examples

```
me = Object.GetMe()

-- Localized in Object Editor as "WARNING!"
title   = Object.GetLocalizedText( me, "WARNING_POPUP_TITLE" )
-- Localized in Object Editor as "Something has gone wrong!"
message = Object.GetLocalizedText( me, "WARNING_POPUP_MESSAGE" )

OsdBasicPopup.SetTitle( popup, title )
OsdBasicPopup.SetMessage( popup, message )
```

See Also

- Object.GetLocalizedText
- Object.GetMe

OsdBasicSingleGraphic

OsdBasicSingleGraphic.SetBackColorType

Brief

Set the background color type for a basic single graphic object.

Definition

```
OsdBasicSingleGraphic.SetBackColorType( OsdBasicSingleGraphic graphicObject, enum colorType )
```

Arguments

graphicObject - The basic single graphic for which to set the background color.colorType - An enum representing the color type. Can be one of the following:
OsdColor.Black
OsdColor.White
OsdColor.Grey
OsdColor.Green
OsdColor.Yellow
OsdColor.Red
OsdColor.Normal
OsdColor.Background
OsdColor.OppositeBg
OsdColor.Foreground
OsdColor.HighlightFg

Return Values

None.

Description

Set the background color type for a basic single graphic object.

Examples

```
OsdBasicSingleGraphic.SetBackColorType( graphicObject, OsdColor.Background )
```

See Also

- OsdBasicSingleGraphic.SetFrontColorType

OsdBasicSingleGraphic.SetBackTexture

Brief

Set the background texture on an OsdBasicSingleGraphic object.

Definition

```
OsdBasicSingleGraphic.SetBackTexture( OsdBasicSingleGraphic graphic, Texture texture )
```

Arguments

graphic - The OsdBasicSingleGraphic object to affect.texture - The Texture object representing the texture to set as the back texture.

Return Values

None.

Description

Set the background texture on an OsdBasicSingleGraphic object.

Examples

```
texture = Texture.Find( "user_interface_back_image" )
OsdBasicSingleGraphic.SetBackTexture( graphic, texture )
```

See Also

- OsdBasicSingleGraphic.SetFrontTexture
- Texture.Find

OsdBasicSingleGraphic.SetFrontColorType

Brief

Set the foreground color type for a basic single graphic object.

Definition

```
OsdBasicSingleGraphic.SetFrontColorType( OsdBasicSingleGraphic graphicObject, enum colorType )
```

Arguments

graphicObject - The basic single graphic for which to set the foreground color.colorType - An enum representing the color type. Can be one of the following:
OsdColor.Black
OsdColor.White
OsdColor.Grey
OsdColor.Green
OsdColor.Yellow
OsdColor.Red
OsdColor.Normal
OsdColor.Background
OsdColor.OppositeBg
OsdColor.Foreground
OsdColor.HighlightFg

Return Values

None.

Description

Set the foreground color type for a basic single graphic object.

Examples

```
OsdBasicSingleGraphic.SetFrontColorType( graphicObject, OsdColor.Foreground )
```

See Also

- OsdBasicSingleGraphic.SetBackColorType

OsdBasicSingleGraphic.SetFrontTexture

Brief

Set the foreground texture on an OsdBasicSingleGraphic object.

Definition

```
OsdBasicSingleGraphic.SetFrontTexture( OsdBasicSingleGraphic graphic, Texture texture )
```

Arguments

graphic - The OsdBasicSingleGraphic object to affect.texture - The Texture object representing the texture to set as the front texture.

Return Values

None.

Description

Set the foreground texture on an OsdBasicSingleGraphic object. The texture must be a local resource of the script's object.

Examples

```
texture = Texture.Find( "user_interface_front_image" )
OsdBasicSingleGraphic.SetFrontTexture( graphic, texture )
```

See Also

- OsdBasicSingleGraphic.SetBackTexture
- Texture.Find

OsdBasicText

OsdBasicText.GetText

Brief

Get the text string stored in a text object.

Definition

```
string OsdBasicText.GetText( OsdBasicText textObject )
```

Arguments

textObject - The OsdBasicText object to query.

Return Values

The text string.

Description

Get the text string stored in a text object.

Examples

```
print( OsdBasicText.GetText( textObject ) )
```

See Also

None

OsdBasicText.SetClipping

Brief

Set the horizontal clipping are for an OsdBasicText object.

Definition

```
OsdBasicText.SetClipping( OsdBasicText textObject, number minClip, number maxClip )
```

Arguments

textObject - The OsdBasicText object to affect.minClip - An offset from the textObject's local align point representing the minimum clipping value.maxClip - An offset from the textObject's local align point representing the maximum clipping value.

Return Values

None.

Description

Set the horizontal clipping are for an OsdBasicText object.

Examples

```
OsdBasicText.SetClipping( textObject, -100, 100 )
```

See Also

None

OsdBasicText.SetFont

Brief

Set the font on an OsdBasicText object.

Definition

```
OsdBasicText.SetFont( OsdBasicText textObject, string font )
```

Arguments

textObject - The OsdBasicText object to affect.font - String representing the font. Can be one of the following:
"13reg"
"13bold"
"18reg"
"18bold"

"26reg"
"26bold"
"60bold"

Return Values

None.

Description

Set the font on an OsdBasicText object.

Examples

```
OsdBasicText.SetFont( textObject, "13reg" )
```

See Also

None

OsdBasicText.SetIgnoreSpecialCharacters

Brief

Set whether or not to ignore special characters.

Definition

```
OsdBasicText.SetIgnoreSpecialCharacters( OsdBasicText textObject, boolean setIgnore )
```

Arguments

textObject - The OsdBasicText object to affect.setIgnore - true if you want to ignore special characters, false otherwise.

Return Values

None.

Description

Set whether or not to ignore special characters. An example of ignoring special characters would be to not convert the substring "[CROSS]" to the graphic representing the cross-button of the PlayStation(R)3 controller.

Examples

```
OsdBasicText.SetIgnoreSpecialCharacters( textObject, true )
```

See Also

None

OsdBasicText.SetScrolling

Brief

Set the scrolling behaviour are for an OsdBasicText object.

Definition

```
OsdBasicText.SetScrolling( OsdBasicText textObject, string scrollType, number scrollSpeed, number
scrollDelay )
```

Arguments

textObject - The OsdBasicText object to affect.scrollType - String representing the scrolling type.scrollSpeed - Scroll speed in pixels per second.scrollDelay - Scroll delay in seconds.

Return Values

None.

Description

Set the scrolling behaviour are for an OsdBasicText object. Valid scrollType strings are:
"none" - no scrolling
"onecycle" - scroll from start to end and back once
"repeatcycle" - scroll from start to end and back and repeat
"nolimit" - scroll "infinitely" in one direction... text will loop and there may be more than once instance of the text visible, depending on text-size vs clipping-size
If scrollType is "nolimit" (ie potentially multiple instances of the text), scrollDelay is the distance (in pixels) between instances of the text.

Examples

```
OsdBasicText.SetScrolling( textObject, "repeatcycle", 20, 1 )
```

See Also

None

OsdBasicText.SetText

Brief

Set the text on an OsdBasicText object.

Definition

```
OsdBasicText.SetText( OsdBasicText textObject, string text )
```

Arguments

textObject - The OsdBasicText object to affect.text - Localized text string.

Return Values

None.

Description

Set the text on an OsdBasicText object.

Examples

```
me = Object.GetMe()

-- Localized in Object Editor as "Hello World!"
message = Object.GetLocalizedText( me, "GREETING" )

OsdBasicText.SetText( textObject, message )
```

See Also

- Object.GetLocalizedText
- Object.GetMe

OsdObject

OsdObject.Activate

Brief

Activate an OSD object.

Definition

```
OsdObject.Activate( OsdObject object, number fadeTime )
```

Arguments

object - The OSD object to activate.fadeTime - The time (in seconds) taken to fade the object in.

Return Values

None.

Description

The specified OSD object is activated, and all descendents of it are activated as well. When an OSD object is first created, it must be activated for it to be displayed. The specified fade time controls the length of time the OsdObject and all of its children take to fade into view. Note that certain OsdObject types update themselves every frame. However, this update function will only be called if the object is activated. For example, a scrolling text object will only scroll while it is activated. In order to toggle the visibility of individual children of an OsdObject, use the function OsdObject.SetVisible.

Examples

```
OsdObject.Activate( object, 1.0 )
```

See Also

- OsdObject.Deactivate
- OsdObject.SetVisible

OsdObject.CreateChildObject

Brief

Creates a new OSD object as a child of the specified parent.

Definition

```
variant OsdObject.CreateChildObject( OsdObject parentObject, string objectType, string childName )
```

Arguments

parentObject - The parent object.objectType - A string specifying the type of the child. Can either be a registered type or an XML file.childName - Name of the new child (optional - defaults to nil)childUid - Uid of the new child (optional - defaults to invalid)beforeUid - Uid of child to insert the new child before (optional - defaults to invalid)setSelectable - Set this child to be selectable (optional - defaults to false)

Return Values

The new child OsdObject-derived object.

Description

Creates a new OSD object as a child of the specified parent. While the name and uid for the new child object are optional, it is necessary to provide atleast one of them if you ever want to be be able to retrieve the child object after creation (assuming you're not storing the reference returned by this method). Note: Garbage collection will not destroy OSD objects. OSD objects are only destroyed either when the parent OSD object is destroyed, or by explicitly calling OsdObject.DestroyObject (or other similar Destroy functions).

Examples

```
childObject = OsdObject.CreateChildObject( parentObject, "OsdBasicChip", "mychild" )
```

See Also

None

OsdObject.Deactivate

Brief

Deactivate an OSD object.

Definition

```
OsdObject.Deactivate( OsdObject object, number fadeTime )
```

Arguments

object - The OSD object to deactivate.fadeTime - The time (in seconds) taken to fade the object out.

Return Values

None.

Description

The specified OSD object is deactivated, and all descendents of it are deactivated as well. The specified fade time controls the length of time the OsdObject and all of its children take to fade from view. Note that certain OsdObject types update themselves every frame. However, this update function will only be called if the object is activated. For example, a scrolling text object will only scroll while it is activated.

Examples

```
OsdObject.Deactivate( object, 1.0 )
```

See Also

- OsdObject.Activate
- OsdObject.SetVisible

OsdObject.DestroyObject

Brief

Destroys an OSD object.

Definition

```
OsdObject.DestroyObject( OsdObject object )
```

Arguments

object - Object to destroy.

Return Values

None.

Description

Destroys an OSD object. Also recursively destroys all children. Note: Garbage collection will not destroy OSD objects. OSD objects are only destroyed either when the parent OSD object is destroyed, or by explicitly calling OsdObject.DestroyObject (or other similar Destroy functions).

Examples

```
OsdObject.DestroyObject( object )
```

See Also

None

OsdObject.FindObject

Brief

Finds an OSD object by name.

Definition

```
OsdObject OsdObject.FindObject( OsdObject currObject, string longName )
```

Arguments

currObject - The object to which the long name is relative.longName - A string specifying a dos/unix-style path of names to the object.

Return Values

The OSD object found.

Description

Finds an OSD object by name.

Examples

```
-- Example 1: Find the child called "banner" of thisObject
foundObject = OsdObject.FindObject( thisObject, "banner" )

-- Example 2: Find the child called "title" of the child called "banner" of thisObject
foundObject = OsdObject.FindObject( thisObject, "banner/title" )

-- Example 3: Find the parent of thisObject
foundObject = OsdObject.FindObject( thisObject, ".." )

-- Example 4: Find the child called "legend" of parent of thisObject, i.e. the sibling called "legend"
of thisObject
foundObject = OsdObject.FindObject( thisObject, "../legend" )
```

See Also

None

OsdObject.GetColorValue

Brief

Get the color of an OSD object.

Definition

```
Vector4 OsdObject.GetColorValue( OsdObject object, Vector4 color )
```

Arguments

object - The OSD object to affect.color - A Vector4 to receive the color (RGBA).

Return Values

None.

Description

Get the color of an OSD object.

Examples

```
local v = Vector4.Create()
OsdObject.GetColorValue( object, v )
print( v )
```

See Also

- OsdObject.SetColorType
- OsdObject.SetColorValue

OsdObject.GetHeight

Brief

Get the height of an OSD object.

Definition

```
number OsdObject.GetHeight( OsdObject object )
```

Arguments

object - The OSD object to query.

Return Values

The height of the object.

Description

Get the height of an OSD object.

Examples

```
local height = OsdObject.GetHeight( object )
```

See Also

None

OsdObject.GetPositionX

Brief

Get the x-component of an object's screen position.

Definition

```
number OsdObject.GetPositionX( OsdObject object )
```

Arguments

object - Object to query.

Return Values

The x-component of the object's screen position.

Description

Get the x-component of an object's screen position.

Examples

```
local x = OsdObject.GetPositionX( object )
```

See Also

None

OsdObject.GetPositionY

Brief

Get the y-component of an object's screen position.

Definition

```
number OsdObject.GetPositionY( OsdObject object )
```

Arguments

object - Object to query.

Return Values

The y-component of the object's screen position.

Description

Get the y-component of an object's screen position.

Examples

```
local y = OsdObject.GetPositionY( object )
```

See Also

None

OsdObject.GetUniqueId

Brief

Get the unique Id of an OSD object.

Definition

```
number|nil OsdObject.GetUniqueId( OsdObject object )
```

Arguments

object - The OSD object to query.

Return Values

The unique id of the object, or nil if one has not been set.

Description

The unique id must be an integer >= 0. The id must be unique with the group of sibling objects.

Examples

```
local uid = OsdObject.GetUniqueId( object )
```

See Also

- OsdObject.SetUniqueId

OsdObject.GetWidth

Brief

Get the width (in parent-space) of an OSD object.

Definition

```
number OsdObject.GetWidth( OsdObject object )
```

Arguments

object - The OSD object to query.

Return Values

The width (in parent-space) of the object.

Description

Get the width (in parent-space) of an OSD object

Examples

```
local width = OsdObject.GetWidth( object )
```

See Also

OsdObject.HandleInput

Brief

Pass input to the OSD object.

Definition

```
boolean OsdObject.HandleInput( OsdObject parent, number inputId, number analogValue=1.0, number
elapsedTime=0.0 )
```

Arguments

inputId - An OsdInput enumeration representing the input type:

- OsdInput.Up
- OsdInput.Down
- OsdInput.Left
- OsdInput.Right
- OsdInput.Accept
- OsdInput.CancelanalogValue - The analog value of the input. Ranges from 0 to 1.elapsedTime - How long (in seconds) the input is being applied for.

Return Values

boolean - true if the input was "consumed", false otherwise.

Description

Pass input to the OSD object. Valid input ids are OsdInput enumeration values. The input is consumed if the OSD object handling it has decided that it should not be handled by anything else.

Examples

```
OsdObject.HandleInput( OsdInput.Accept )
```

See Also

None

OsdObject.IsActive

Brief

Query whether or not an OSD object is active (activated).

Definition

```
boolean OsdObject.IsActive( OsdObject object )
```

Arguments

object - The OSD object to query.

Return Values

true if the OSD object is activated, false otherwise.

Description

Query whether or not an OSD object is active (activated).

Examples

```
if OsdObject.IsActive( object ) then
    print( "OSD object is active!" )
end
```

See Also

None

OsdObject.SetAlphaType

Brief

Set the alpha type of an OSD object.

Definition

```
OsdObject.SetAlphaType( OsdObject object, enum alphaType )
```

Arguments

object - The OSD object to affectalphaType - An enum representing the alpha type. Can be one of the following:
OsdAlpha.Normal
OsdAlpha.Ninety
OsdAlpha.ThreeQuarter
OsdAlpha.Half

OsdAlpha.Quarter
OsdAlpha.Eighth
OsdAlpha.Sixteenth

Return Values

None.

Description

Set the alpha type of an OSD object.

Examples

```
OsdObject.SetAlphaType( object, OsdAlpha.Normal )
```

See Also

None

OsdObject.SetAlphaValue

Brief

Set the alpha value of an OSD object.

Definition

```
OsdObject.SetAlphaValue( OsdObject object, number alphaValue )
```

Arguments

object - The OSD object to affect.alphaValue - A number between 0 (fully transparent) and 1 (fully opaque).

Return Values

None.

Description

Set the alpha value of an OSD object.

Examples

```
OsdObject.SetAlphaValue( object, 0.5 )
```

See Also

None

OsdObject.SetAngle

Brief

Set the angle of an OSD object.

Definition

```
OsdObject.SetAngle( OsdObject object, number angle )
```

Arguments

object - The OSD object to affect.angle - The angle to set.

Return Values

None.

Description

Set the angle of an OSD object.

Examples

```
OsdObject.SetAngle( object, 10.0 )
```

See Also

None

OsdObject.SetColorType

Brief

Set the color type of an OSD object.

Definition

```
OsdObject.SetColorType( OsdObject object, enum colorType )
```

Arguments

object - The OSD object to affect.colorType - An enum representing the color type. Can be one of the following:
OsdColor.Black
OsdColor.White
OsdColor.Grey
OsdColor.Green
OsdColor.Yellow
OsdColor.Normal
OsdColor.Background
OsdColor.OppositeBg
OsdColor.Foreground
OsdColor.HighlightFg

Return Values

None.

Description

Set the color type of an OSD object.

Examples

```
OsdObject.SetColorType( object, OsdColor.Foreground )
```

See Also

None

OsdObject.SetColorValue

Brief

Set the color of an OSD object.

Definition

```
OsdObject.SetColorValue( OsdObject object, Vector4 color )
```

Arguments

object - The OSD object to affect.color - A Vector4 representing the color (RGBA).

Return Values

None.

Description

Set the color of an OSD object. Setting the color in this way (instead of OsdObject.SetColorType) has an extra memory and processing overhead.

Examples

```
OsdObject.SetColorValue( object, Vector4.Create( 1.0, 0.0, 0.0, 1.0 ) )
```

See Also

- OsdObject.SetColorType

OsdObject.SetHeight

Brief

Set the height of an OSD object.

Definition

```
OsdObject.SetHeight( OsdObject object, number height )
```

Arguments

object - The OSD object to affect.height - The height to set.

Return Values

None.

Description

Set the height of an OSD object.

Examples

```
OsdObject.SetHeight( object, 100.0 )
```

See Also

None

OsdObject.SetHeightFromChildren

Brief

Set the height of an OSD object so as to bound all children.

Definition

```
OsdObject.SetHeightFromChildren( OsdObject object, number border )
```

Arguments

object - The OSD object to affect.border - Extra border to add to height.

Return Values

None.

Description

Set the height of an OSD object so as to bound all children.

Examples

```
OsdObject.SetHeightFromChildren( width, 10.0 )
```

See Also

None

OsdObject.SetLocalSpaceAlignPoint

Brief

Set the local-space align point of an OSD object.

Definition

```
OsdObject.SetLocalSpaceAlignPoint( OsdObject object, enum alignPoint )
```

Arguments

object - The OSD object to affect.alignPoint - An enum representing the align point. Can be one of the following:

- OsdAlign.Center
- OsdAlign.Top
- OsdAlign.Bottom
- OsdAlign.Left
- OsdAlign.Right
- OsdAlign.TopLeft
- OsdAlign.TopRight

- OsdAlign.BottomLeft
- OsdAlign.BottomRight

Return Values

None.

Description

Set the local-space align point of an OSD object.

Examples

```
OsdObject.SetLocalSpaceAlignPoint( object, OsdAlign.Center )
```

See Also

None

OsdObject.SetName

Brief

Set the name of an object.

Definition

```
OsdObject.SetName( OsdObject object, string name )
```

Arguments

object - The object to set the name of.name - The new name for the object.

Return Values

None.

Description

Set the name of an object. The name should be unique in the set of siblings.

Examples

```
OsdObject.SetName( object, "banner" )
```

See Also

None

OsdObject.SetParentSpaceAlignPoint

Brief

Set the parent-space align-point of an OSD object.

Definition

```
OsdObject.SetParentSpaceAlignPoint( OsdObject object, enum alignPointOrigin, number alignPointOffsetX
= 0.0, number alignPointOffsetY = 0.0 )
```

Arguments

object - The OSD object to affect.alignPointOrigin - An enum representing the align point origin. Can be one of the following:

- OsdAlign.Center
- OsdAlign.Top
- OsdAlign.Bottom
- OsdAlign.Left
- OsdAlign.Right
- OsdAlign.TopLeft
- OsdAlign.TopRight
- OsdAlign.BottomLeft
- OsdAlign.BottomRightalignPointOffsetX - A number representing the x-component of the align point offset.alignPointOffsetY - A number representing the y-component of the align point offset.

Return Values

None.

Description

An OSD object is positioned relative to its parent (or, if its parent is "intangible", its next tangible "ancestor"). If a parent has no tangible ancestors, it is positioned in screen-space. An OSD object's position is specified as an offset from a point in its parent; that point is called the "parent-space align-point origin" and the offset is called the "parent-space align-point offset". The resultant position is called the "parent-space align-point". A second align point, the "local-space align-point", is also defined. This point is within the object itself. The object is finally positioned be placing the local-space align-point over the parent-space align-point. Both the parent-space align-point origin and the local-space align-point are defined as one of the following relative points within the parent and child object respectively: center top bottom left right top-left top-right bottom-left bottom-right

Examples

```
OsdObject.SetParentSpaceAlignPoint( object, OsdAlign.TopLeft, 100, 200 )
```

See Also

- OsdObject.SetLocalSpaceAlignPoint

OsdObject.SetPosition

Brief

Sets the position of an object.

Definition

```
OsdObject.SetPosition( OsdObject object, enum parentSpaceAlignPointOrigin, number
parentSpaceAlignPointOffsetX = 0.0, number parentSpaceAlignPointOffsetY = 0.0, enum
localSpaceAlignPoint = nil )
```

Arguments

object - The object to set the position of.parentSpaceAlignPointOrigin - An OsdAlign enum representing the origin of the parent-space align point.parentSpaceAlignPointOffsetX - The x-offset of the parent-space align point.parentSpaceAlignPointOffsetY - The y-offset of the parent-space align point.localSpaceAlignPoint - An OsdAlign enum representing the local-space align point.

Return Values

None.

Description

Sets the position of an object. The position is set relative to the object's (tangible) parent, or relative to the screen if no (tangible) parent exists. The position is defined by lining up two points; one in the parent-space and one in the local-space. The parent-space point is defined by an origin and an offset. If the local-space align point is omitted, it is taken to be the same as the parent-space align point origin.

Examples

```
OsdObject.SetPosition( object, OsdAlign.TopLeft, 100.0, 200.0, OsdAlign.Bottom )
```

See Also

None

OsdObject.SetScale

Brief

Set the scale of an OSD object.

Definition

```
OsdObject.SetScale( OsdObject object, number scale )
```

Arguments

object - The OSD object to affect.scale - The scale to set.

Return Values

None.

Description

Set the scale of an OSD object.

Examples

```
OsdObject.SetScale( object, 2.0 )
```

See Also

None

OsdObject.SetUniqueId

Brief

Set the unique Id of an OSD object.

Definition

```
OsdObject.SetUniqueId( OsdObject object, number uid = nil )
```

Arguments

object - The OSD object.uid - The unique id.

Return Values

None.

Description

The unique id must be an integer >= 0 or nil. The id must be unique with the group of sibling objects. Passing nil in clears the id.

Examples

```
OsdObject.SetUniqueId( object, 1 )
```

See Also

- OsdObject.GetUniqueId

OsdObject.SetVisible

Brief

Set an OSD object to be visible or invisible.

Definition

```
OsdObject.SetVisible( OsdObject object, boolean isVisible = true )
```

Arguments

object - OSD object to affect.isVisible - true or false (defaults to true).

Return Values

None.

Description

The specified OSD object is made visible or invisible, and all descendents of it inherit the same visibility setting. Use this function to show or hide specific components of an OSD interface element. For example, individual menu items of an OsdBasicMenu can be shown or hidden. When a new OsdObject is created, it is initially in a deactivated state. Once it is activated, it will be visible until it is made invisible with this function or deactivated. Invisible objects are still updated. For example, a scrolling text object will continue to scroll even when invisible.

Examples

```
OsdObject.SetVisible( object, false )
```

See Also

- OsdObject.Activate
- OsdObject.Deactivate

OsdObject.SetWidth

Brief

Set the width of an OSD object.

Definition

```
OsdObject.SetWidth( OsdObject object, number width )
```

Arguments

object - The OSD object to affect.width - The width to set.

Return Values

None.

Description

Set the width of an OSD object.

Examples

```
OsdObject.SetWidth( width, 200.0 )
```

See Also

None

OsdObject.SetWidthFromChildren

Brief

Set the width of an OSD object so as to bound all children.

Definition

```
OsdObject.SetWidthFromChildren( OsdObject object, number border )
```

Arguments

object - The OSD object to affect.border - Extra border to add to width.

Return Values

None.

Description

Set the width of an OSD object so as to bound all children.

Examples

```
OsdObject.SetWidthFromChildren( width, 10.0 )
```

See Also

None

OsdSelectionParent

OsdSelectionParent.BeginChanges

Brief

Call before making multiple changes to the selection parent.

Definition

```
OsdSelectionParent.BeginChanges( OsdSelectionParent parent )
```

Arguments

parent - The parent to affect.

Return Values

None.

Description

Call before making multiple changes to the selection parent. For example, adding/disabling/enabling children.

Examples

```
OsdSelectionParent.BeginChanges( parent )
```

See Also

None

OsdSelectionParent.EndChanges

Brief

Call after making changes to the selection parent.

Definition

```
OsdSelectionParent.EndChanges( OsdSelectionParent parent )
```

Arguments

parent - The parent to affect.

Return Values

None.

Description

Call after making changes to the selection parent. For example, adding children.

Examples

```
OsdSelectionParent.EndChanges( parent )
```

See Also

None

OsdSelectionParent.GetCurrentSelectionId

Brief

Returns the id of the currently selected child of the selection parent.

Definition

```
number OsdSelectionParent.GetCurrentSelectionId( OsdSelectionParent parent )
```

Arguments

parent - The OsdSelectionParent object.

Return Values

The id of the currently selected object.

Description

Returns the id of the currently selected child of the selection parent. The ids are defined by id(X) in the init string of the child.

Examples

```
local selected = OsdSelectionParent.GetCurrentSelectionId( parentObject )
```

See Also

None

OsdSelectionParent.SetObjectEnabled

Brief

Enable or disable a child of the selection parent object.

Definition

```
OsdSelectionParent.SetObjectEnabled( OsdSelectionParent parent, OsdObject child, boolean wantEnabled )
```

Arguments

parent - The OsdSelectionParent object.child - The child to enable/disable.wantEnabled - true or false.

Return Values

None.

Description

Enable or disable a child of the selection parent object. Disabled children are removed from the selection process.

Examples

```
OsdSelectionParent.SetObjectEnabled( parent, child, false )
```

See Also

None

OsdSimpleMenu

OsdSimpleMenu.AddItem

Brief

Add a simple menu item to a simple menu object.

Definition

```
number OsdSimpleMenu.AddItem( OsdSimpleMenu self, string title, number id, boolean setEnabled )
```

Arguments

self - The menu object for which to call the method.title - Text to display on the menu item.id - Non-negative unique (in this menu) identifier for the menu item, a negative value causes the item to be allocated an id by the system.setEnabled - true to have the menu item enabled (selectable) initially, false otherwise.

Return Values

The id of the menu item.

Description

Add a simple menu item to a simple menu object. The item will be added to the bottom of the menu. AddItem should be called between calls to BeginChanges and EndChanges.

Examples

```
me = Object.GetMe()

-- Text localized in Object Editor as "Play Game"
title1 = Object.GetLocalizedText( me, "PLAY_MENU_TITLE" )
-- Text localized in Object Editor as "About Game"
title2 = Object.GetLocalizedText( me, "ABOUT_MENU_TITLE" )

menuObject = OsdObject.CreateChildObject( parentObject, "OsdSimpleMenu", "menu" )
OsdSimpleMenu.BeginChanges( menuObject )
OsdSimpleMenu.AddItem( menuObject, title1, 0, true )
OsdSimpleMenu.AddItem( menuObject, title2, 1, true )
OsdSimpleMenu.EndChanges( menuObject )
```

See Also

- Object.GetLocalizedText
- Object.GetMe

OsdSimpleMenu.SetItemTitle

Brief

Change the text displayed on a simple menu item.

Definition

```
OsdSimpleMenu.SetItemTitle( OsdSimpleMenu self, number id, string title )
```

Arguments

self - The menu object for which to call the method.title - Text to display on the menu item.id - Unique identifier for the menu item.

Return Values

None.

Description

Change the text displayed on a simple menu item. The text will have been initially set when the item was added to the menu.

Examples

```
me = Object.GetMe()

-- Text localized in Object Editor as "Play Game"
title1 = Object.GetLocalizedText( me, "PLAY_MENU_TITLE" )
-- Text localized in Object Editor as "About Game"
title2 = Object.GetLocalizedText( me, "ABOUT_MENU_TITLE" )
-- Text localized in Object Editor as "End Game"
title3 = Object.GetLocalizedText( me, "END_MENU_TITLE" )

menuObject = OsdObject.CreateChildObject( parentObject, "OsdSimpleMenu", "menu" )
OsdSimpleMenu.BeginChanges( menuObject )
OsdSimpleMenu.AddItem( menuObject, title1, 0, true )
OsdSimpleMenu.AddItem( menuObject, title2, 1, true )
OsdSimpleMenu.EndChanges( menuObject )
OsdSimpleMenu.SetItemTitle( menuObject, 1, title3 )
```

See Also

- Object.GetLocalizedText
- Object.GetMe

OsdSkinnedBox

OsdSkinnedBox.ResetTextures

Brief

Reset the textures.

Definition

```
OsdSkinnedBox.ResetTextures( OsdSkinnedBox box )
```

Arguments

box - The OsdSkinnedBox object.

Return Values

None.

Description

Reset the textures. This will set all textures to nil, and is equivalent to passing nil to all arguments of SetTextures.

Examples

```
box = OsdObject.CreateChildObject( Osd.GetRoot(), "OsdSkinnedBox", "test" )
OsdSkinnedBox.SetTextures( box, Texture.Find("t1"), Texture.Find("t2"), Texture.Find("t3"),
Texture.Find("t4"), Texture.Find("t5"), Texture.Find("t6"), Texture.Find("t7"), Texture.Find("t8"),
Texture.Find("t9") )
OsdSkinnedBox.ResetTextures( box )
```

See Also

- OsdSkinnedBox.SetTextures

OsdSkinnedBox.SetColorGradientEnabled

Brief

Enable/disable the color gradient.

Definition

```
OsdSkinnedBox.SetColorGradientEnabled( OsdSkinnedBox box, boolean enable = true )
```

Arguments

box - The OsdSkinnedBox object.enable - true to enable the color gradient, false otherwise.

Return Values

None.

Description

Enable/disable the color gradient. The color gradient is applied from top to bottom and the parameters of it cannot be adjusted. The colour is based on the current OSD object colour, transitioning to either a lighter or darker shade, depending on the additive/subtractive setting.

Examples

```
box = OsdObject.CreateChildObject( Osd.GetRoot(), "OsdSkinnedBox", "test" )
OsdSkinnedBox.SetTextures( box, Texture.Find("t1"), Texture.Find("t2"), Texture.Find("t3"),
Texture.Find("t4"), Texture.Find("t5"), Texture.Find("t6"), Texture.Find("t7"), Texture.Find("t8"),
Texture.Find("t9") )
OsdSkinnedBox.SetColorGradientEnabled( box, true )
```

See Also

- OsdSkinnedBox.SetColorGradientSubtractive
- OsdSkinnedBox.SetColorGradientReversed
- OsdSkinnedBox.SetColorGradientMagnitude

OsdSkinnedBox.SetColorGradientMagnitude

Brief

Set the magnitude of the gradient.

Definition

```
OsdSkinnedBox.SetColorGradientMagnitude( OsdSkinnedBox box, number magnitude )
```

Arguments

box - The OsdSkinnedBox object.magnitude - The gradient magnitude.

Return Values

None.

Description

Set the magnitude of the gradient. Valid gradient magnitudes are in the range zero to one.

Examples

```
box = OsdObject.CreateChildObject( Osd.GetRoot(), "OsdSkinnedBox", "test" )
OsdSkinnedBox.SetTextures( box, Texture.Find("t1"), Texture.Find("t2"), Texture.Find("t3"),
Texture.Find("t4"), Texture.Find("t5"), Texture.Find("t6"), Texture.Find("t7"), Texture.Find("t8"),
Texture.Find("t9") )
OsdSkinnedBox.SetColorGradientEnabled( box, true )
OsdSkinnedBox.SetColorGradientMagnitude( box, 0.5 )
```

See Also

- OsdSkinnedBox.SetColorGradientEnabled
- OsdSkinnedBox.SetColorGradientSubtractive
- OsdSkinnedBox.SetColorGradientReversed

OsdSkinnedBox.SetColorGradientReversed

Brief

Set whether or not the direction of the color gradient is reversed.

Definition

```
OsdSkinnedBox.SetColorGradientReversed( OsdSkinnedBox box, boolean reversed = true )
```

Arguments

box - The OsdSkinnedBox object.reversed - true to set the gradient reversed, false otherwise.

Return Values

None.

Description

Set whether or not the direction of the color gradient is reversed.

Examples

```
box = OsdObject.CreateChildObject( Osd.GetRoot(), "OsdSkinnedBox", "test" )
OsdSkinnedBox.SetTextures( box, Texture.Find("t1"), Texture.Find("t2"), Texture.Find("t3"),
Texture.Find("t4"), Texture.Find("t5"), Texture.Find("t6"), Texture.Find("t7"), Texture.Find("t8"),
Texture.Find("t9") )
OsdSkinnedBox.SetColorGradientEnabled( box, true )
OsdSkinnedBox.SetColorGradientReversed( box, true )
```

See Also

- OsdSkinnedBox.SetColorGradientEnabled
- OsdSkinnedBox.SetColorGradientSubtractive
- OsdSkinnedBox.SetColorGradientMagnitude

OsdSkinnedBox.SetColorGradientSubtractive

Brief

Set whether or not the color gradient is subtractive.

Definition

```
OsdSkinnedBox.SetColorGradientSubtractive( OsdSkinnedBox box, boolean subtractive = true )
```

Arguments

box - The OsdSkinnedBox object.subtractive - true to set the gradient subtractive, false otherwise.

Return Values

None.

Description

Set whether or not the color gradient is subtractive. If it is not subtractive, then it is additive. An additive gradient will transition to a lighter shade, subtractive to a darker shade.

Examples

```
box = OsdObject.CreateChildObject( Osd.GetRoot(), "OsdSkinnedBox", "test" )
OsdSkinnedBox.SetTextures( box, Texture.Find("t1"), Texture.Find("t2"), Texture.Find("t3"),
Texture.Find("t4"), Texture.Find("t5"), Texture.Find("t6"), Texture.Find("t7"), Texture.Find("t8"),
Texture.Find("t9") )
OsdSkinnedBox.SetColorGradientEnabled( box, true )
OsdSkinnedBox.SetColorGradientSubtractive( box, true )
```

See Also

- OsdSkinnedBox.SetColorGradientEnabled
- OsdSkinnedBox.SetColorGradientReversed
- OsdSkinnedBox.SetColorGradientMagnitude

OsdSkinnedBox.SetTexture

Brief

Set a texture.

Definition

```
OsdSkinnedBox.SetTexture( OsdSkinnedBox box, enum positionId, Texture t = nil)
```

Arguments

box - The OsdSkinnedBox object.positionId - An enum representing which texture to set. Can be one of:

- OsdAlign.TopLeft
- OsdAlign.Top
- OsdAlign.TopRight
- OsdAlign.Left
- OsdAlign.Center
- OsdAlign.Right
- OsdAlign.BottomLeft
- OsdAlign.Bottom
- OsdAlign.BottomRightt - The texture.

Return Values

None.

Description

Set a texture.

Examples

```
box = OsdObject.CreateChildObject( Osd.GetRoot(), "OsdSkinnedBox", "test" )
OsdSkinnedBox.SetTextures( box, Texture.Find("t1"), Texture.Find("t2"), Texture.Find("t3"),
Texture.Find("t4"), Texture.Find("t5"), Texture.Find("t6"), Texture.Find("t7"), Texture.Find("t8"),
Texture.Find("t9") )
OsdSkinnedBox.SetTexture( box, OsdAlign.Center, Texture.Find("alt") )
```

See Also

- OsdSkinnedBox.SetTextures

OsdSkinnedBox.SetTextures

Brief

Set the textures.

Definition

```
OsdSkinnedBox.SetTextures( OsdSkinnedBox box, Texture t1, Texture t2, Texture t3, Texture t4, Texture
t5, Texture t6, Texture t7, Texture t8, Texture t9 )
```

Arguments

box - The OsdSkinnedBox object.t1 - The top-left texture, or nil to clear.t2 - The top-center texture, or nil to clear.t3 - The top-right texture, or nil to clear.t4 - The left-center texture, or nil to clear.t5 - The center texture, or nil to clear.t6 - The right-center texture, or nil to clear.t7 - The bottom-left texture, or nil to clear.t8 - The bottom-center texture, or nil to clear.t9 - The bottom-right texture, or nil to clear.

Return Values

None.

Description

Set the textures for the skinned box. Nil can be passed to any texture, which will hide that part of the skinned box.

Examples

```
box = OsdObject.CreateChildObject( Osd.GetRoot(), "OsdSkinnedBox", "test" )
OsdSkinnedBox.SetTextures( box, Texture.Find("t1"), Texture.Find("t2"), Texture.Find("t3"),
Texture.Find("t4"), Texture.Find("t5"), Texture.Find("t6"), Texture.Find("t7"), Texture.Find("t8"),
Texture.Find("t9") )
```

See Also

- OsdSkinnedBox.SetTexture
- OsdSkinnedBox.ResetTextures

OsdTextLines

OsdTextLines.GetNumTextLines

Brief

Get the number of lines of text for an OsdTextLines object.

Definition

```
number OsdTextLines.GetNumTextLines( OsdTextLines textLinesObject )
```

Arguments

textLinesObject - OsdTextLines to query.

Return Values

The number of lines of text.

Description

Get the number of lines of text for an OsdTextLines object.

Examples

```
numLines = OsdTextLines.GetNumTextLines( textLinesObject )
```

See Also

None

OsdTextLines.SetFont

Brief

Set the font on an OsdTextLines object.

Definition

```
OsdTextLines.SetFont( OsdTextLines textLinesObject, string font )
```

Arguments

textLinesObject - The OsdTextLines object to affect.font - String representing the font. Can be one of the following:
"13reg"
"13bold"
"18reg"
"18bold"
"26reg"
"26bold"
"60bold"

Return Values

None.

Description

Set the font on an OsdTextLines object.

Examples

```
OsdTextLines.SetFont( textLinesObject, "13reg" )
```

See Also

None

OsdTextLines.SetJustification

Brief

Set the justification on an OsdTextLines object.

Definition

```
OsdTextLines.SetJustification( OsdTextLines textLinesObject, enum justify )
```

Arguments

textLinesObject - The OsdTextLines object to affect.justify - An enum representing the justification. Can be one of the following:
OsdAlign.Left
OsdAlign.Right
OsdAlign.Center
OsdAlign.LeftRight

Return Values

None.

Description

Set the justification on an OsdTextLines object.

Examples

```
OsdTextLines.SetJustification( textLinesObject, OsdAlign.Center )
```

See Also

None

OsdTextLines.SetLineSpacing

Brief

Set the line spacing on an OsdTextLines object.

Definition

```
OsdTextLines.SetLineSpacing( OsdTextLines textLinesObject, number lineSpacing )
```

Arguments

textLinesObject - The OsdTextLines object to affect.lineSpacing - The distance between lines of text.

Return Values

None.

Description

Set the line spacing on an OsdTextLines object.

Examples

```
OsdTextLines.SetLineSpacing( textLinesObject, 10 )
```

See Also

None

OsdTextLines.SetText

Brief

Set the text on an OsdTextLines object.

Definition

```
OsdTextLines.SetText( OsdTextLines textLinesObject, string localizedText )
```

Arguments

textLinesObject - The OsdTextLines object to affect.localizedText - Localized text string.

Return Values

None.

Description

Set the text on an OsdTextLines object.

Examples

```
me = Object.GetMe()

-- Localized in Object Editor as "Hello World!"
message = Object.GetLocalizedText( me, "GREETING" )

OsdTextLines.SetText( textLinesObject, message )
```

See Also

- Object.GetLocalizedText
- Object.GetMe

OsdTextPanel

OsdTextPanel.GetTextBorder

Brief

Get the size of the border around the text for an OsdTextPanel object.

Definition

```
number OsdTextPanel.GetTextBorder( OsdTextPanel textPanelObject )
```

Arguments

textPanelObject - The OsdTextPanel object to query.

Return Values

The size of the border around the text.

Description

Get the size of the border around the text for an OsdTextPanel object.

Examples

```
border = OsdTextPanel.GetTextBorder( textPanelObject )
```

See Also

None

OsdTextPanel.GetTextLines

Brief

Get the internal textlines object from an OsdTextPanel object.

Definition

```
OsdTextLines OsdTextPanel.GetTextLines( OsdTextPanel textPanelObject )
```

Arguments

textPanelObject - The OsdTextPanel object to query.

Return Values

The OsdTextLines object.

Description

Get the internal textlines object from an OsdTextPanel object.

Examples

```
textLinesObject = OsdTextPanel.GetTextLines( textPanelObject )
```

See Also

None

OsdTextPanel.IsFitHeightToText

Brief

Query whether an OsdTextPanel object's height is adjusted to its text.

Definition

```
boolean OsdTextPanel.IsFitHeightToText( OsdTextPanel textPanelObject )
```

Arguments

textPanelObject - The OsdTextPanel object to query.

Return Values

true or false.

Description

Query whether an OsdTextPanel object's height is adjusted to its text.

Examples

```
OsdTextPanel.IsFitHeightToText( textPanelObject )
```

See Also

None

OsdTextPanel.IsPanelVisible

Brief

Query whether an OsdTextPanel object's background panel is visible.

Definition

```
boolean OsdTextPanel.IsPanelVisible( OsdTextPanel textPanelObject )
```

Arguments

textPanelObject - The OsdTextPanel object to query.

Return Values

true or false.

Description

Query whether an OsdTextPanel object's background panel is visible.

Examples

```
if OsdTextPanel.IsPanelVisible( textPanelObject ) then
    print( "Panel is visible!" )
end
```

See Also

None

OsdTextPanel.IsScrollBarVisible

Brief

Query whether an OsdTextPanel object's scroll-bar is visible.

Definition

```
boolean OsdTextPanel.IsScrollBarVisible( OsdTextPanel textPanelObject )
```

Arguments

textPanelObject - The OsdTextPanel object to query.

Return Values

true or false.

Description

Query whether an OsdTextPanel object's scroll-bar is visible.

Examples

```
OsdTextPanel.IsScrollBarVisible( textPanelObject )
```

See Also

None

OsdTextPanel.SetFitHeightToText

Brief

Set whether or not you want the height of the text panel to adjust to the text.

Definition

```
OsdTextPanel.SetFitHeightToText( OsdTextPanel textPanelObject, boolean isFitHeightToText )
```

Arguments

textPanelObject - The OsdTextPanel object to affect.isFitHeightToText - Boolean value (true or false).

Return Values

None.

Description

Set whether or not you want the height of the text panel to adjust to the text.

Examples

```
OsdTextPanel.SetFitHeightToText( textPanelObject, true )
```

See Also

None

OsdTextPanel.SetFont

Brief

Set the font on an OsdTextPanel object.

Definition

```
OsdTextPanel.SetFont( OsdTextPanel textPanelObject, string font )
```

Arguments

textPanelObject - The OsdTextPanel object to affect.font - String representing the font. Can be one of the following:
"13reg"
"13bold"
"18reg"
"18bold"
"26reg"
"26bold"
"60bold"

Return Values

None.

Description

Set the font on an OsdTextPanel object.

Examples

```
OsdTextPanel.SetFont( textPanelObject, "13reg" )
```

See Also

None

OsdTextPanel.SetJustification

Brief

Set the justification on an OsdTextPanel object.

Definition

```
OsdTextPanel.SetJustification( OsdTextPanel textPanelObject, enum justify )
```

Arguments

textPanelObject - The OsdTextPanel object to affect.justify - An enum representing the justification. Can be one of the following:

- OsdAlign.Left
- OsdAlign.Right
- OsdAlign.Center
- OsdAlign.LeftRight

Return Values

None.

Description

Set the justification on an OsdTextPanel object.

Examples

```
OsdTextPanel.SetJustification( textPanelObject, OsdAlign.Center )
```

See Also

None

OsdTextPanel.SetLineSpacing

Brief

Set the line spacing on an OsdTextPanel object.

Definition

```
OsdTextPanel.SetLineSpacing( OsdTextPanel textPanelObject, number lineSpacing )
```

Arguments

textPanelObject - The OsdTextPanel object to affect.lineSpacing - The distance between lines of text.

Return Values

None.

Description

Set the line spacing on an OsdTextPanel object.

Examples

```
OsdTextPanel.SetLineSpacing( textPanelObject, 10 )
```

See Also

None

OsdTextPanel.SetPanelVisible

Brief

Set whether or not the panel background is visible.

Definition

```
OsdTextPanel.SetPanelVisible( OsdTextPanel textPanelObject, boolean isPanelVisible )
```

Arguments

textPanelObject - The OsdTextPanel object to affect.isPanelVisible - Boolean value (true or false).

Return Values

None.

Description

Set whether or not the panel background is visible.

Examples

```
OsdTextPanel.SetPanelVisible( textPanelObject, true )
```

See Also

None

OsdTextPanel.SetScrollBarVisible

Brief

Set whether or not the scroll bar is visible.

Definition

```
OsdTextPanel.SetScrollBarVisible( OsdTextPanel textPanelObject, boolean isScrollBarVisible )
```

Arguments

textPanelObject - The OsdTextPanel object to affect.isScrollBarVisible - Boolean value (true or false).

Return Values

None.

Description

Set whether or not the scroll bar is visible.

Examples

```
OsdTextPanel.SetScrollBarVisible( textPanelObject, false )
```

See Also

None

OsdTextPanel.SetScrollType

Brief

Set the scroll-type on an OsdTextPanel object.

Definition

```
OsdTextPanel.SetScrollType( OsdTextPanel textPanelObject, enum scrollType )
```

Arguments

textPanelObject - The OsdTextPanel object to affect.scrollType - An OsdMotion enum representing the scroll-type. Can be one of the following:
OsdMotion.None
OsdMotion.ConstSpeed
OsdMotion.Spring

Return Values

None.

Description

Set the scroll-type on an OsdTextPanel object.

Examples

```
OsdTextPanel.SetScrollType( textPanelObject, "smooth" )
```

See Also

None

OsdTextPanel.SetText

Brief

Set the text of a OsdTextPanel object.

Definition

```
OsdTextPanel.SetText(OsdTextPanel textPanelObject, string text, number maxTextLength = -1, boolean
trimLeadingChars = false)
```

Arguments

textPanelObject - The OsdTextPanel object to affect.text - The text.maxTextLength - A positive value to limit the text length, or -1 otherwise.trimLeadingChars - (If the text is too long) true, to trim leading character, false to trim trailing characters.

Return Values

None.

Description

Set the text of a OsdTextPanel object. The maxTextLength parameter can be used to limit the number of characters copied into the object. It actually limits the number of bytes, so if the string contains multibyte characters, there may be fewer actual characters than specified. If the function needs to remove characters to comply with maxTextLength, the characters will be removed from the end unless trimLeadingChars is true, in which case the characters will be removed from the beginning.

Examples

```
me = Object.GetMe()

-- Text localized in Object Editor
panelText = Object.GetLocalizedText( me, "PANEL_MESSAGE" )

OsdTextPanel.SetText( textPanelObject, panelText )
```

See Also

- Object.GetLocalizedText
- Object.GetMe

OsdTextPanel.SetTextBorder

Brief

Set the size of the border around the text on an OsdTextPanel object.

Definition

```
OsdTextPanel.SetTextBorder( OsdTextPanel textPanelObject, number textBorder )
```

Arguments

textPanelObject - The OsdTextPanel object to affect.textBorder - The text border size.

Return Values

None.

Description

Set the size of the border around the text on an OsdTextPanel object.

Examples

```
OsdTextPanel.SetTextBorder( textPanelObject, 10 )
```

See Also

None

OsdTextPanel.SetTextColorType

Brief

Set the color of the text on an OsdTextPanel object.

Definition

```
OsdTextPanel.SetTextColorType( OsdTextPanel textPanelObject, enum colorType )
```

Arguments

textPanelObject - The OsdTextPanel object to affect.colorType - An enum representing the color type. Can be one of the following:
OsdColor.Black
OsdColor.White
OsdColor.Grey
OsdColor.Green
OsdColor.Yellow
OsdColor.Red
OsdColor.Normal
OsdColor.Background
OsdColor.OppositeBg
OsdColor.Foreground
OsdColor.HighlightFg

Return Values

None.

Description

Set the color of the text on an OsdTextPanel object.

Examples

```
OsdTextPanel.SetTextColorType( textPanelObject, OsdColor.Foreground )
```

See Also

None

Osk

Osk.Close

Brief

Closes the OSK opened by this script.

Definition

```
Osk.Close()
```

Arguments

None

Return Values

None

Description

Closes the OSK opened by this script. If the OSK was not opened by this script, then this function has no effect.

Examples

```
Osk.Close()
```

See Also

- Osk.Open

Osk.GetText

Brief

Obtain the text the user entered in the OSK.

Definition

```
string Osk.GetText()
```

Arguments

None

Return Values

The text that was entered by the user or nil if the user cancelled input or not in the callback context.

Description

Returns the text that was entered by the user in the OSK. Only valid during the registered callback for the OSK request that triggered the call.

Examples

```
mytext = Keyboard.GetText()
```

See Also

None

Osk.IsAvailable

Brief

Checks if it is possible to to open the OSK.

Definition

```
boolean Osk.IsAvailable()
```

Arguments

None

Return Values

true if the script is able to open the OSK, false if the OSK is unavailable.

Description

Checks if it is possible to to open the OSK.

Examples

```
if Osk.IsAvailable() == true then
    Osk.Open( "OnText()", "How old are you?", "", 3, false, true, false )
end
```

See Also

None

Osk.Open

Brief

Opens the OSK to request a string from the user.

Definition

```
boolean Osk.Open( function callback, string title, string initialText, number maxCharacters,
CharacterSet characterSet, boolean allowReturn )
```

Arguments

callback - The Lua function to execute when the OSK has a result. Max 127 characters.title - The title to give the OSK. Max 127 characters.initialText - The initial text to add to the OSK. Max 511 characters.maxCharacters - The maximum number of characters (not bytes) to request from the user. Max 511 characters.characterSet - A valid character set enum value (see details)allowReturn - Set to true to allow the user to enter new line characters.

Return Values

false if the OSK could not be started at this time.

Description

Opens up an instance of the OSK to accept text input from the user. Only one instance of the OSK can be open at a time. characterSet can be any of the following:

- CharacterSet.Default
- CharacterSet.LatinOnly
- CharacterSet.NumericalOnly
- CharacterSet.Password

Examples

```
success = Osk.Open( OnNumerical, "What is your pin?", "0000", 4, CharacterSet.NumericalOnly, false )
```

See Also

None

OutboundMessage

OutboundMessage.Create

Brief

Creates a new OutboundMessage for sending.

Definition

```
OutboundMessage OutboundMessage.Create( number messageType, number bufferSize )
```

Arguments

messageType - Message type as a number. This is free for the script to set and is sent as an int32.bufferSize - Size of the buffer to write the message to. This should be equal to the size of the data you want to include in the message.

Return Values

A new instance of an OutboundMessage.

Description

Creates a new OutboundMessage for sending. The size for the buffer will be allocated out of the script's network environment (currently the global network pool).

Examples

```
local message = OutboundMessage.Create( 12345, 16 )
```

See Also

None

OutboundMessage.GetRemainingBufferSize

Brief

Returns the space remaining in the message's buffer.

Definition

```
number OutboundMessage.GetRemainingBufferSize( OutboundMessage msg )
```

Arguments

msg - The instance of the message to query.

Return Values

The remaining buffer space in bytes.

Description

Returns the space remaining in the message's buffer.

Examples

```
local size = OutboundMessage.GetRemainingBufferSize( message )
```

See Also

None

OutboundMessage.Reset

Brief

Resets the buffer for the OutboundMessage.

Definition

```
OutboundMessage.Reset( OutboundMessage msg )
```

Arguments

msg - The instance of the message to reset.

Return Values

None.

Description

Resets the message buffer for the OutboundMessage so that it can be used again with new contents.

Examples

```
local message = OutboundMessage.Create( 0, 4 )
OutboundMessage.WriteInt16( message, value1 )
OutboundMessage.WriteInt16( message, value2 )
OutboundMessage.Send( message, personId )

-- Reuse message
OutboundMessage.Reset( message )
OutboundMessage.WriteInt32( message, value3 )
OutboundMessage.Send( message )
```

See Also

- OutboundMessage.Create
- OutboundMessage.Send
- OutboundMessage.WriteInt32

OutboundMessage.Send

Brief

Sends the OutboundMessage.

Definition

```
OutboundMessage.Send( OutboundMessage msg, string personId = nil )
```

Arguments

msg - The instance of the message to send.personId - The person ID to send the message to. If it is not specified the message is sent to all clients.

Return Values

None.

Description

Sends the specified OutboundMessage. The message can be sent to a specific person or broadcast to all persons located in the current scene.

Examples

```
local message = OutboundMessage.Create( 0, 6 )
OutboundMessage.WriteInt16( message, index )
OutboundMessage.WriteInt32( message, score )

personId = Person.GetId( otherPerson )
OutboundMessage.Send( message, personId )
```

See Also

None

OutboundMessage.SendUnguaranteed

Brief

Sends the OutboundMessage over UDP.

Definition

```
OutboundMessage.SendUnguaranteed( OutboundMessage msg, string personId = nil )
```

Arguments

msg - The instance of the message to send.personId - The person ID to send the message to. If it is not specified the message is sent to all clients.

Return Values

None.

Description

Sends the specified OutboundMessage. The message can be sent to a specific person or broadcast to all persons located in the current scene. Please note that UDP messages are unguaranteed, out-of-order. Care must be taken when using this transport type as it is not suitable for all

message. Do not use this if your message must reach the target user. If ordering is essential, this can be achieved by simply adding a time stamp to the message and manually checking and discarding out-of-order messages on the receiving side. The global function GetInstanceTime can be used as one method of manually adding a timestamp.

Examples

```
local message = OutboundMessage.Create( 0, 6 )
OutboundMessage.WriteInt16( message, index )
OutboundMessage.WriteInt32( message, score )

personId = Person.GetId( otherPerson )
OutboundMessage.SendUnguaranteed( message, personId )
```

See Also

- GetInstanceTime
- OutboundMessage.Send

OutboundMessage.WriteBigInt

Brief

Write a BigInt value into an outbound message.

Definition

```
OutboundMessage.WriteBigInt( OutboundMessage msg, BigInt bigInt )
```

Arguments

msg - The message.bigInt - The BigInt value to write.

Return Values

None.

Description

Writes a BigInt value into an outbound message. Any size of BigInt is permissible.

Examples

```
local message = OutboundMessage.Create(0, 8 )
local bigInt = BigInt.Create(64, "0x1234567890ABCDEF")
OutboundMessage.WriteBigInt( message, bigInt )
OutboundMessage.Send( message )
```

See Also

- BigInt.Create

OutboundMessage.WriteFloat

Brief

Writes a 32-bit floating point value (float) to the message.

Definition

```
OutboundMessage.WriteFloat( OutboundMessage msg, number data )
```

Arguments

msg - The instance of the message to write to.data - 32-bit floating point value to write to the message packet.

Return Values

None.

Description

Writes a 32-bit floating point value (float) to the message.

Examples

```
local message = OutboundMessage.Create( 0, 4 )
OutboundMessage.WriteFloat( message, 3.14159 )
OutboundMessage.Send( message )
```

See Also

- OutboundMessage.WriteInt8
- OutboundMessage.WriteInt16
- OutboundMessage.WriteInt32
- OutboundMessage.WriteString

OutboundMessage.WriteInt8

Brief

Writes an 8-bit integer to the message.

Definition

```
OutboundMessage.WriteInt8( OutboundMessage msg, number data )
```

Arguments

msg - The instance of the message to write to.data - 8-bit integer to write to the message packet.

Return Values

None.

Description

Writes an 8-bit integer (or byte) to the message.

Examples

```
local message = OutboundMessage.Create( 0, 1 )
OutboundMessage.WriteInt8( message, value )
OutboundMessage.Send( message )
```

See Also

- OutboundMessage.WriteInt16
- OutboundMessage.WriteInt32
- OutboundMessage.WriteFloat
- OutboundMessage.WriteString

OutboundMessage.WriteInt16

Brief

Writes a 16-bit integer (short) to the message.

Definition

```
OutboundMessage.WriteInt16( OutboundMessage msg, number data )
```

Arguments

msg - The instance of the message to write to.data - 16-bit integer to write to the message packet.

Return Values

None.

Description

Writes a 16-bit integer (short) to the message.

Examples

```
local message = OutboundMessage.Create( 0, 2 )
OutboundMessage.WriteInt16( message, index )
OutboundMessage.Send( message )
```

See Also

- OutboundMessage.WriteInt8
- OutboundMessage.WriteInt32
- OutboundMessage.WriteFloat
- OutboundMessage.WriteString

OutboundMessage.WriteInt32

Brief

Writes a 32-bit integer to the message.

Definition

```
OutboundMessage.WriteInt32( OutboundMessage msg, number data )
```

Arguments

msg - The instance of the message to write to.data - 32-bit integer to write to the message packet.

Return Values

None.

Description

Writes a 32-bit integer to the message.

Examples

```
local message = OutboundMessage.Create( 0, 4 )
OutboundMessage.WriteInt32( message, index )
OutboundMessage.Send( message )
```

See Also

- OutboundMessage.WriteInt8
- OutboundMessage.WriteInt16
- OutboundMessage.WriteFloat
- OutboundMessage.WriteString

OutboundMessage.WriteString

Brief

Writes a string value to the message.

Definition

```
OutboundMessage.WriteString( OutboundMessage msg, string data )
```

Arguments

msg - The instance of the message to write to.data - string value to write to the message packet.

Return Values

None.

Description

Writes a string value to the message.

Examples

```
greeting = "Hello World!"
-- Msg size is length of string plus 1 for terminating character
msgSize  = string.len( greeting ) + 1
local message = OutboundMessage.Create( 0, msgSize )
OutboundMessage.WriteString( message, greeting )
OutboundMessage.Send( message )
```

See Also

- OutboundMessage.WriteInt8

- OutboundMessage.WriteInt16
- OutboundMessage.WriteInt32
- OutboundMessage.WriteFloat

Pad

Pad.CreatePads

Brief

Creates the specified number of pads.

Definition

```
Pad.CreatePads( number padCount = 1 )
```

Arguments

padCount - The number of pads required. Valid range is 1 to 4.

Return Values

None.

Description

Creates the specified number of pads required by the script. This should never be called more than once.

Examples

```
-- Create a single pad
Pad.CreatePads()
pad = Pad.GetPad()
```

See Also

- Pad.GetPad

Pad.GetAnalogExtent

Brief

Returns the analog value of the stick, button or sensor.

Definition

```
number Pad.GetAnalogExtent( Pad pad, number actionId )
```

Arguments

pad - The pad to query.actionId - The user defined action ID corresponding to a button ID.

Return Values

The analogue value of the controller stick, button or sensor.

Description

Returns the analog value of the stick, button or motion sensor in the following ranges: Analogue stick: -1.0 to 1.0. Analogue button: 0.0 (not pressed) to 1.0 (fully pressed). Motion sensor: -1.0 to 1.0. Note: For the motion sensor, if the controller is stationary, gravity will still cause the returned value to be non-zero (depending on the angle at which the controller is held).

Examples

```
-- Initialization code
Pad.CreatePads()
pad = Pad.GetPad()
TURN_LEFT = 5
Pad.Reserve( pad, PAD_DPAD_LEFT, TURN_LEFT )

function OnUpdate()
    print( "Button value = " ..  Pad.GetAnalogExtent( pad, myLeft ) )
end
```

See Also

- Pad.CreatePads
- Pad.GetPad
- Pad.Reserve

Pad.GetPad

Brief

Gets the specified pad object.

Definition

```
Pad Pad.GetPad( number which = 1 )
```

Arguments

which - The pad to get. Ranges from 1 to the number of pads created.

Return Values

The pad corresponding to the specified index or nil on failure.

Description

Gets the specified pad object.

Examples

```
-- Create two pads
Pad.CreatePads( 2 )
pad1 = Pad.GetPad( 1 )
pad2 = Pad.GetPad( 2 )
```

See Also

- Pad.CreatePads

Pad.GetSensorForwards

Brief

Returns the vector for the forwards direction of the pad.

Definition

```
Vector4 Pad.GetSensorForwards( Pad pad )
```

Arguments

pad - The pad to query.

Return Values

The vector for the forwards direction of the pad.

Description

Returns the vector for the forwards direction of the pad.

Examples

```
local direction = Pad.GetSensorForwards( pad )
```

See Also

- Pad.GetSensorRight
- Pad.GetSensorUp

Pad.GetSensorRight

Brief

Returns the vector for the right direction of the pad.

Definition

```
Vector4 Pad.GetSensorRight( Pad pad )
```

Arguments

pad - The pad to query.

Return Values

The vector for the right direction of the pad.

Description

Returns the vector for the right direction of the pad.

Examples

```
local direction = Pad.GetSensorRight( pad )
```

See Also

- Pad.GetSensorUp
- Pad.GetSensorForwards

Pad.GetSensorUp

Brief

Returns the vector for the up direction of the pad.

Definition

```
Vector4 Pad.GetSensorUp( Pad pad )
```

Arguments

pad - The pad to query.

Return Values

The vector for the up direction of the pad.

Description

Returns the vector for the up direction of the pad.

Examples

```
local direction = Pad.GetSensorUp( pad )
```

See Also

- Pad.GetSensorRight
- Pad.GetSensorForwards

Pad.IsCircleAccept

Brief

Returns true if the pad associates PAD_CIRCLE with PAD_ACCEPT and false otherwise.

Definition

```
boolean Pad.IsCircleAccept( Pad pad )
```

Arguments

pad - The pad to query.

Return Values

Returns true if the pad associates PAD_CIRCLE with PAD_ACCEPT and false otherwise.

Description

Returns true if the pad associates PAD_CIRCLE with PAD_ACCEPT and false otherwise.

Examples

```
Pad.CreatePads()
pad = Pad.GetPad()
if( Pad.IsCircleAccept( pad ) ) then
    print( "Pad uses circle for accept" );
else
    print( "Pad uses cross for accept" );
end
```

See Also

- Pad.CreatePads
- Pad.GetPad

Pad.IsConnected

Brief

Returns true if the pad is connected and false otherwise.

Definition

```
boolean Pad.IsConnected( Pad pad )
```

Arguments

pad - The pad to query.

Return Values

true if the pad is connected and false otherwise.

Description

Returns true if the pad is connected and false otherwise.

Examples

```
Pad.CreatePads()
pad = Pad.GetPad()
print( "Pad connected = " .. Pad.IsConnected( pad ) )
```

See Also

- Pad.CreatePads
- Pad.GetPad

Pad.IsHeld

Brief

Detects whether the specified reserved pad button is currently pressed.

Definition

```
boolean Pad.IsHeld( Pad pad, number actionId )
```

Arguments

pad - The pad to query.actionId - The user defined action ID corresponding to a button ID

Return Values

true if the button is held and false otherwise.

Description

Detects whether the reserved pad button associated with the specified action ID is currently pressed.

Examples

```
-- Initialization code
Pad.CreatePads()
pad = Pad.GetPad()
FIRE_LASER = 5
Pad.Reserve( pad, PAD_CROSS, FIRE_LASER )

function OnUpdate()
    if Pad.IsHeld( pad, FIRE_LASER ) then
        print( "Laser firing sustained blast!" )
    end
end
```

See Also

- Pad.CreatePads
- Pad.GetPad
- Pad.Reserve

Pad.Reserve

Brief

Makes a controller function queryable.

Definition

```
Pad.Reserve( Pad pad, number buttonId, number actionId = buttonId, boolean repeat = false, boolean
dropThrough = false )
```

Arguments

pad - The pad to affect.buttonId - The controller function to reserve. Can be one of the following:
PAD_DPAD_UP
PAD_DPAD_DOWN
PAD_DPAD_LEFT
PAD_DPAD_RIGHT
PAD_JOY_LEFT_X
PAD_JOY_LEFT_Y
PAD_JOY_RIGHT_X
PAD_JOY_RIGHT_Y
PAD_ACCEPT
PAD_SQUARE
PAD_TRIANGLE

PAD_DECLINE
PAD_L1_SHOULDER
PAD_L2_SHOULDER
PAD_L3_JOY
PAD_R1_SHOULDER
PAD_R2_SHOULDER
PAD_R3_JOY
PAD_CROSS
PAD_CIRCLE
PAD_SENSOR_AXIS_X
PAD_SENSOR_AXIS_Y
PAD_SENSOR_AXIS_Z
PAD_SENSOR_GYRO_Y
PAD_ACTUATOR_SMALL
PAD_ACTUATOR_BIGactionId - The user defined action ID to map the button ID to.repeat - If set to true, the Pad.WasJustPressed function can return true for a single button press which is held down continuously.dropThrough - If set to true, other input handlers with lower priority can also receive the button event. Normally only the highest priority input handler with the button reserved will receive the button event.

Return Values

None.

Description

Reserves the specified controller function and assigns an abstract action ID to use for that button.

Examples

```
-- Initialization code
Pad.CreatePads()
pad = Pad.GetPad()
FIRE_LASER = 5
Pad.Reserve( pad, PAD_CROSS, FIRE_LASER )
```

See Also

- Pad.UnReserve

Pad.SetActuator

Brief

Sets the specified actuator's vibration strength.

Definition

```
boolean Pad.SetActuator( Pad pad, number actuator, number value )
```

Arguments

pad - The pad to affect.actuator - The actuator to set. Can be one of the following:
PAD_ACTUATOR_SMALL
PAD_ACTUATOR_BIGvalue - The value to set the actuator to. Valid values for PAD_ACTUATOR_SMALL are only 0 or 1,
for PAD_ACTUATOR_BIG it is all values from 0.0 to 1.0.

Return Values

true if sucessfully set and false otherwise.

Description

Sets the specified actuator's vibration strength. If called by multiple scripts or the same script multiple times per frame it accumulates the values. The actuator must be reserved before it can be set. The small actuator can only be on or off, the rotation frequency cannot be changed.

Examples

```
-- Initialization code
Pad.CreatePads()
pad = Pad.GetPad()
Pad.Reserve( pad, PAD_ACTUATOR_BIG )
Pad.Reserve( pad, PAD_ACTUATOR_SMALL )

function OnUpdate()
    if ( smallEarthquake == true ) then
        Pad.SetActuator( pad, PAD_ACTUATOR_SMALL, 0.5)
    elseif ( bigEarthquake == true ) then
        Pad.SetActuator( pad, PAD_ACTUATOR_BIG, 1.0)
    end
end
```

See Also

- Pad.CreatePads
- Pad.GetPad
- Pad.Reserve

Pad.UnReserve

Brief

Relinquishes the previously reserved action ID.

Definition

```
Pad.UnReserve( Pad pad, number actionId )
```

Arguments

pad - The pad to affect.actionId - The action ID to unreserve.

Return Values

None.

Description

Relinquishes the previously reserved action ID.

Examples

```
-- Initialization code
Pad.CreatePads()
pad = Pad.GetPad()
FIRE_LASER = 5
Pad.Reserve( pad, PAD_CROSS, FIRE_LASER )

-- Termination code
Pad.UnReserve( pad, FIRE_LASER )
```

See Also

- Pad.CreatePads

- Pad.GetPad
- Pad.Reserve

Pad.UnReserveAll

Brief

Relinquishes any previously reserved action IDs.

Definition

```
Pad.UnReserveAll(Pad pad)
```

Arguments

pad - The pad to affect.

Return Values

None.

Description

Relinquishes any previously reserved action IDs.

Examples

```
Pad.UnReserveAll(pad)
```

See Also

- Pad.Reserve
- Pad.UnReserve

Pad.WasAnyJustPressed

Brief

Detects whether any reserved pad button was pressed.

Definition

```
boolean Pad.WasAnyJustPressed( Pad pad )
```

Arguments

pad - The pad to query.

Return Values

true if any reserved button was just pressed and false otherwise.

Description

Detects whether any reserved pad button was pressed.

Examples

```
-- Initialization code
Pad.CreatePads()
pad = Pad.GetPad()
FIRE_LASER = 5
FIRE_MINIGUN = 6
Pad.Reserve( pad, PAD_SQUARE, FIRE_LASER )
Pad.Reserve( pad, PAD_CIRCLE, FIRE_MINIGUN )

function OnUpdate()
    if Pad.WasAnyJustPressed( pad ) then
        print( "A weapon was fired!" )
    end
end
```

See Also

- Pad.CreatePads
- Pad.GetPad
- Pad.Reserve

Pad.WasJustPressed

Brief

Detects whether the specified reserved pad button was pressed.

Definition

```
boolean Pad.WasJustPressed( Pad pad, number actionId )
```

Arguments

pad - The pad to query.actionId - The user defined action ID corresponding to a button ID.

Return Values

true if the button was pressed and false otherwise.

Description

Detects whether the reserved pad button associated with the specified action ID was pressed. Note that analogue sticks moving outside their dead zone counts as a button press.

Examples

```
-- Initialization code
Pad.CreatePads()
pad = Pad.GetPad()
FIRE_LASER = 5
Pad.Reserve( pad, PAD_CROSS, FIRE_LASER )

function OnUpdate()
    if Pad.WasJustPressed( pad, FIRE_LASER ) then
        print( "Laser fired successfully!" )
    end
end
```

See Also

- Pad.CreatePads
- Pad.GetPad
- Pad.Reserve

Pad.WasJustReleased

Brief

Detects whether the specified reserved pad button was released.

Definition

```
boolean Pad.WasJustReleased( Pad pad, number actionId )
```

Arguments

pad - The pad to query.actionId - The user defined action ID corresponding to a button ID

Return Values

true if the button was just released and false otherwise.

Description

Detects whether the reserved pad button associated with the specified action ID was released. Note that analogue sticks returning to their dead zone counts as a button release.

Examples

```
-- Initialization code
Pad.CreatePads()
pad = Pad.GetPad()
FIRE_LASER = 5
Pad.Reserve( pad, PAD_CROSS, FIRE_LASER )

function OnUpdate()
    if Pad.WasJustReleased( pad, FIRE_LASER ) then
        print( "Laser deactivated!" )
    end
end
```

See Also

- Pad.CreatePads
- Pad.GetPad
- Pad.Reserve

Person

Person.AttachToEntity

Brief

Attaches the person to specified entity so when the entity moves around the person moves with it.

Definition

```
Person.AttachToEntity( Person person, Entity entity = nil, Matrix44 localTransform = nil )
```

Arguments

entity - the entity to attach to or nil to detach player from current entity.

Return Values

None.

Description

When attached to an entity, setting position and rotation will affect the local transform only. The transform must be locked by the calling lua environment otherwise a lua error is generated. If the person is unlocked it automatically detaches from the entity.

Examples

```
person = LocalPlayer.GetPerson()
car = Entity.Create()
Person.AttachToEntity( person, car )
```

See Also

None

Person.DoAction

Brief

Plays an action on a Person

Definition

```
Person.DoAction( Person person, string actionName, boolean autoUnlock = false )
Person.DoAction( Person person, string actionName, boolean autoUnlock, number unlockFeatures )
```

Arguments

person - the person to play an action on.actionName - Name of the action to be performed. It can be a custom animation loaded remotely through LocalPlayer.AddRepertoire as well as one of the following standard actions:
"agree"
"beckon"
"bow"
"cheer"
"clap"
"disagree"
"frustration"
"laugh"
"point"
"wave"
"stand_up_from_seat"autoUnlock - if true, then the animation lock on the person is automatically released after the action has completed. Defaults to false.unlockFeatures - only used if the autoUnlock parameter is true. If omitted then defaults to releasing the locks that were set when Person.DoAction was called, otherwise will only release the locks specified.
features specified.

Return Values

Returns true if the action succeeds.

Description

The animation must be locked by the calling lua environment otherwise a lua error is generated. If the action isn't recognized a lua error is generated. If it fails and autoUnlock is true it unlocks immediately.

Examples

```
if Person.HasLock( person ) then
    Person.DoAction( person, "clap" )
end
```

See Also

- Person.Lock
- Person.Unlock
- Person.IsLocked
- Person.HasLock
- Person.SetBehavior
- Person.DoActionBroadcast

Person.DoActionBroadcast

Brief

Plays an action on a Person

Definition

```
Person.DoActionBroadcast( Person person, string actionName, boolean autoUnlock = false )
Person.DoActionBroadcast( Person person, string actionName, boolean autoUnlock, number unlockFeatures
)
```

Arguments

person - the person to play an action on.actionName - Name of the action to be performed. It can be a custom animation loaded remotely through LocalPlayer.AddRepertoire as well as one of the following standard actions:
"agree"
"beckon"
"bow"
"cheer"
"clap"
"disagree"
"frustration"
"laugh"
"point"
"wave"
"stand_up_from_seat"autoUnlock - if true, then the animation lock on the person is automatically released after the action has completed. Defaults to false.unlockFeatures - only used if the autoUnlock parameter is true. If omitted then defaults to releasing the locks that were set when Person.DoActionBroadcast was called, otherwise will only release the locks specified.
features specified.

Return Values

Returns true if the action succeeds.

Description

The animation must be locked by the calling lua environment otherwise a lua error is generated. If the action isn't recognized a lua error is

generated. If it fails and autoUnlock is true it unlocks immediately. If calling DoActionBroadcast on the LocalPlayer, this function will broadcast the action to other clients in the scene. Using DoActionBroadcast impacts on the amount of network traffic, so avoid calling this function frequently.

Examples

```
if Person.HasLock( person ) then
    Person.DoActionBroadcast( person, "clap" )
end
```

See Also

- Person.Lock
- Person.Unlock
- Person.IsLocked
- Person.HasLock
- Person.SetBehavior
- Person.DoAction

Person.EnableChatBubbles

Brief

Sets chat bubbles on or off.

Definition

```
Person.EnableChatBubbles( Person person, boolean enabled )
```

Arguments

enabled - True to enable chat bubbles or false to disable.

Return Values

None.

Description

With this function, a script can allow chat bubbles to be switched on or off at will. Chat will still appear in the chat log as normal. Please note: In order to use this function, the is_allowed_chat_bubble_change property in the object's mini_game component must be set to true.

Examples

```
local person = LocalPlayer.GetPerson( );
if ( Person.Lock( person, PersonLockType.ChatBubbles ) == true ) then
    Person.EnableChatBubbles( person, false );
end
```

See Also

- Person.Lock
- Person.Unlock

Person.EnableCollision

Brief

Toggles collision detection for the person's avatar.

Definition

```
Person.EnableCollision( Person person, boolean value )
```

Arguments

person - the person to modify.value - Whether collision should be on or off.

Return Values

None.

Description

Toggles collision detection for the person's avatar. The collision must be locked by the calling lua environment otherwise a lua error is generated.

Examples

```
if Person.IsCollision( person ) then
    Person.EnableCollision( person, false )
end
```

See Also

- Person.IsCollision

Person.FindInInstance

Brief

Finds a person in the instance from the person ID.

Definition

```
Person Person.FindInInstance( string personId )
```

Arguments

personId - The person to find.

Return Values

A Person if found and nil otherwise.

Description

Finds a person in the instance from the specified person ID.

Examples

```
me = Person.FindInInstance( LocalPlayer.GetPersonId() )
```

See Also

- LocalPlayer.GetPersonId

Person.GetAnimReg

Brief

Returns the value of the specified animation register.

Definition

```
number|boolean|string Person.GetAnimReg(Person person, string registerName)
```

Arguments

person - person to query.registerName - the name of the animation register to get.

Return Values

The value of the register, with the type based on the register type.

Description

If the register doesn't exist or the animation register doesn't have Lua read access an error will be thrown.

Examples

```
fatness = person:GetAnimReg("Fatness")
```

See Also

- Person.SetAnimReg

Person.GetAvatarStatusType

Brief

Returns the current avatar status type of the person specified

Definition

```
number Person.GetAvatarStatusId( Person person )
```

Arguments

person - The person to query.

Return Values

The enumerated value of the current status type.

Description

The user can change the status of their avatar from the emotes menu. This method returns the current user status type. Changing the status type affects the facial animation (mood) used, and the status text above the avatar that remote users see. The possible status types are:
AvatarStatusType.Neutral
AvatarStatusType.Happy
AvatarStatusType.Sad
AvatarStatusType.Angry

AvatarStatusType.Confused
AvatarStatusType.Unwell
AvatarStatusType.Busy
AvatarStatusType.IsLookingForHelp
AvatarStatusType.WantToChat
AvatarStatusType.WantToPlayGame

Examples

```
me = LocalPlayer.GetPerson()
if Person.GetAvatarStatusType( me ) == AvatarStatusType.Happy then
    DoSomethingHappy()
end
```

See Also

- Person.GetMood

Person.GetBehavior

Brief

Returns the name of the current behavior playing on the person.

Definition

```
string Person.GetBehavior( Person person )
```

Arguments

person - The person to query.

Return Values

The name of the current behavior playing on the person.

Description

Returns the name of the current behavior playing on the person or nil if the behavior is a Home internal behavior. For a list of Home's public behaviors see Person.SetBehavior. For post 1.40 behaviour, if the behaviour belongs to the calling script environment, will only specify the name. If the behaviour belongs to another object, that has allowed its behaviours to be public, then will specify the object id, as well as the name. For pre-1.40 behaviour. For the local player, will only return the name of the behaviour, but for remote persons, will return the fully qualified name (objectId:name) - this is for backwards compatibility. If the behaviour is a system public behaviour, will return the name only.

Examples

```
behaviorName = Person.GetBehavior( person )
```

See Also

- Person.SetBehavior

Person.GetBehaviorCategory

Brief

Returns the specified person's behavior state.

Definition

```
string|nil Person.GetBehaviorCategory( Person person )
```

Arguments

person - The person to update.

Return Values

Returns the specified person's behavior state.

Description

Returns nil if in an internal behavior or be one of the following: "dancing" "posing" "sitting_on_floor" "sitting_on_seat" "standing" "moving" Behaviors should be catagorized into one of the above states. This function can be used to check if a person is dancing etc. without checking the animation behavior name.

Examples

```
if Person.GetBehaviorCategory( LocalPlayer.GetPerson() ) == 'dancing' then
    StartTheParty()
end
```

See Also

- Person.GetBehavior
- Person.SetBehavior

Person.GetBoneCount

Brief

Get the number of bones in the person's skeleton.

Definition

```
number Person.GetBoneCount(Person person)
```

Arguments

person - Person to query.

Return Values

Number of bones in the skeleton, or 0 if the skeleton is not available.

Description

Retrieves the number of bones/joints in the person's skeleton. If no skeleton is present or loaded, 0 is returned.

Examples

```
local boneCount = LocalPlayer.GetPerson():GetBoneCount()
```

See Also

- Person.GetBoneIndex

Person.GetBoneIndex

Brief

Find the index of a bone in the person's skeleton.

Definition

```
number Person.GetBoneIndex(Person person, string boneName)
```

Arguments

person - Person to query.boneName - Name of bone to locate.

Return Values

The index of the bone in the skeleton (in range 1 - GetBoneCount()), or 0 if not found.

Description

Retrieves the index of the specified bone/joint in the person's skeleton. The index is 1-based. If the bone cannot be found, or there is no skeleton attached or loaded, 0 is returned.

Examples

```
local leftHandIndex = LocalPlayer.GetPerson():GetBoneIndex("lefthand")
```

See Also

- Person.GetBoneCount
- Person.GetBoneLocalMatrix
- Person.GetBoneWorldMatrix

Person.GetBoneLocalMatrix

Brief

Get the local transform matrix for a bone in the skeleton.

Definition

```
Person.GetBoneLocalMatrix(Person person, string|number bone, Matrix44 result)
```

Arguments

person - Person to query.bone - Bone name or index.result (out) - Matrix44 value to store result into.

Return Values

None.

Description

Retrieve the local-space transform matrix for a specified bone in the skeleton. If the bone is not found an error will result. Retrieving the matrix by index will be more efficient if done more than once.

Examples

```
local mat = Matrix44.Create()
LocalPlayer.GetPerson():GetBoneLocalMatrix("lefthand", mat)
-- or
local leftHandIndex = LocalPlayer.GetPerson():GetBoneIndex("lefthand")
LocalPlayer.GetPerson():GetBoneLocalMatrix(leftHandIndex, mat)
```

See Also

- Person.GetBoneWorldMatrix
- Person.GetBoneIndex

Person.GetBoneWorldMatrix

Brief

Get the world transform matrix for a bone in the skeleton.

Definition

```
Person.GetBoneWorldMatrix(Person person, string|number bone, Matrix44 result)
```

Arguments

person - Person to query.bone - Bone name or index.result (out) - Matrix44 value to store result into.

Return Values

None.

Description

Retrieve the world-space transform matrix for a specified bone in the skeleton. If the bone is not found an error will result. Retrieving the matrix by index will be more efficient if done more than once.

Examples

```
local mat = Matrix44.Create()
LocalPlayer.GetPerson():GetBoneWorldMatrix("lefthand", mat)
-- or
local leftHandIndex = LocalPlayer.GetPerson():GetBoneIndex("lefthand")
LocalPlayer.GetPerson():GetBoneWorldMatrix(leftHandIndex, mat)
```

See Also

- Person.GetBoneLocalMatrix
- Person.GetBoneIndex

Person.GetCurrentAction

Brief

Returns the name of the current action the person is in, if any.

Definition

```
string Person.GetCurrentAction( Person person )
```

Arguments

person - The person to query.

Return Values

Returns the name of the current action.

Description

If the person is not in an action nil is returned. Home internal actions will also return nil. For a list of Home public actions see Person.DoAction.

Examples

```
actionName = Person.GetCurrentAction( LocalPlayer.GetPerson() )
```

See Also

- LocalPlayer.DoAction
- LocalPlayer.IsDoingAction

Person.GetGroundUserId

Brief

Return the userID of the collision on which the person is standing.

Definition

```
number Person.GetGroundUserId( Person person )
```

Arguments

person - The person to query

Return Values

The userID.

Description

Return the userID of the collision on which the person is standing. Check Person.IsOnGround before calling this function; if the person is not on the ground an error will occur.

Examples

```
if ( person:IsOnGround() ) then
    print( Person.GetGroundUserId( person ) )
end
```

See Also

- Person.IsOnGround

Person.GetHeight

Brief

Retrieve the avatar's height in meters.

Definition

```
number Person.GetHeight(Person person)
```

Arguments

person - the person to query.

Return Values

The height of the person's avatar in meters, or nil if the avatar is not yet loaded.

Description

The value returned is the height of the avatar when in the bind pose, given the current height settings, in meters. This will return nil if the avatar is currently a fallback.

Examples

```
if ( person:GetHeight() < 1.0 ) then print ("Hello shorty") end
```

See Also

None

Person.GetId

Brief

Returns the person's ID.

Definition

```
string Person.GetId( Person person )
```

Arguments

person - The person to query.

Return Values

The person's ID.

Description

Returns the person's ID. A person's ID value is unique per instance, but not across instances and is not persistent. That is, a person ID can be used within an instance to uniquely identify any person in that instance, but it should be assumed to be assigned at random to players entering the instance, and only valid while that player is in the instance. Players in another instance of the same scene will not be accessible by person ID. Players exiting and re-entering the instance will almost certainly be assigned a different person ID, but a person's ID will not change while they remain in the same instance.

Examples

```
me = LocalPlayer.GetPerson()
if Person.GetId( me ) ~= LocalPlayer.GetPersonId() then
    print( "Oh no!" )
end
```

See Also

- LocalPlayer.GetPerson
- LocalPlayer.GetPersonId

Person.GetIdsOfFriendsInInstance

Brief

Returns a table of people IDs in the current instance that are friends.

Definition

```
string[] Person.GetIdsOfFriendsInInstance()
```

Arguments

None

Return Values

A table of person ID strings.

Description

Returns a table of person IDs in the current instance that are friends.

Examples

```
ids = Person.GetIdsOfFriendsInInstance()
numFriends = #ids
print( "There are " .. numFriends .. " friends in the instance." )
```

See Also

None

Person.GetIdsOfRemotePeopleInInstance

Brief

Returns a table of all remote person IDs in the current instance.

Definition

```
string[] Person.GetIdsOfRemotePeopleInInstance()
```

Arguments

None

Return Values

A table of person ID strings.

Description

Returns a table of person IDs for all remote avatars in the current instance.

Examples

```
ids = Person.GetIdsOfRemotePeopleInInstance()
numAvatars = #ids + 1
print( "There are " .. numAvatars .. " avatars in the instance." )
```

See Also

None

Person.GetInstanceSessionMasterId

Brief

Returns the current instance's session master Person ID.

Definition

```
string Person.GetInstanceSessionMasterId()
```

Arguments

None

Return Values

A person ID of the session master or nil if not ready yet.

Description

Returns the current instance's session master Person ID. Each instance has a session master automatically assigned to it. When a new instance is created the first person in that instance will be the session master and if they leave then a new session master is chosen if other users are in the instance.

Examples

```
local sessionMasterId = Person.GetInstanceSessionMasterId()
local sessionMaster = nil
if sessionMasterId ~= nil then
    sessionMaster = Person.FindInInstance( sessionMasterId )
    SendNetworkMessageToRequestInstanceState( sessionMaster )
end
```

See Also

- User.GetSceneOwner

Person.GetInventoryItemObjectId

Brief

Obtain the object id of the currently deployed companion object.

Definition

```
string Person.GetInventoryItemObjectId(Person person)
```

Arguments

person - The person to query.

Return Values

The object ID of the deployed inventory item if present, and nil otherwise.

Description

Obtains the object id of the currently deployed companion object, or returns nil if one is not currently active.

Examples

```
if (Person.GetInventoryItemObjectId(person) == "12345678-12345678-12345678-12345678") then
    isAllowedIntoMiniGame = true
end
```

See Also

None

Person.GetLocalBounds

Brief

Retrieve the local-space bounds of a person.

Definition

```
boolean Person.GetLocalBounds(Person person, Vector4 min, Vector4 max)
```

Arguments

person - The person to query.min (out) - Vector4 value to receive the minimum extent.max (out) - Vector4 value to receive the maximum extent.

Return Values

True if the bounds were retrieved successfully, false if the person is not yet loaded.

Description

Get the local-space bounds of a person's model. This will reflect the user's clothing choices and height selection. If the person's avatar is not loaded, this function will return false.

Examples

```
local min = Vector4.Create()
local max = Vector4.Create()
if (LocalPlayer.GetPerson():GetLocalBounds(min, max)) then
    print("Local player avatar is", max:Y() - min:Y(), "metres tall")
end
```

See Also

- Person.GetWorldBounds

Person.GetMood

Brief

Returns the mood of the specified person.

Definition

```
MoodType Person.GetMood(Person person)
```

Arguments

person - the person to interrogate the mood value for.

Return Values

The enumerated value of the mood.

Description

The mood type is the facial animation that is used on the idling person. This is controlled by the user through the status option in the emotes menu. Different statuses can share the same mood, ie Confused and IsLookingForHelp statuses both use the same mood Confused. The enumerated type of MoodType is:
MoodType.Neutral
MoodType.Happy
MoodType.Sad
MoodType.Angry
MoodType.Confused

Examples

```
if (person:GetMood() == MoodType.Sad) then
    print("Why the long face?")
end
```

See Also

- Person.GetAvatarStatusType

Person.GetName

Brief

Returns the person's Online ID.

Definition

```
string Person.GetName( Person person )
```

Arguments

person - The person to query.

Return Values

The person's Online ID.

Description

Returns the person's Online ID, but only if you launch HomeDeveloper.self in online mode. If launching in offline mode, only the local player's Person will be accessible, and the function will return the string "Me".

Examples

```
me = LocalPlayer.GetPerson()
print( "My name is " .. Person.GetName( me ) )
```

See Also

- LocalPlayer.GetPerson

Person.GetPosition

Brief

Returns the position of the person.

Definition

```
Vector4 Person.GetPosition( Person person )
Person.GetPosition( Person person, Vector4 outVec )
```

Arguments

person - The person to query.outVec - The Vector4 to store the result to.

Return Values

The position of the person.

Description

Returns a vector containing the position of the specified person.

Examples

```
me = LocalPlayer.GetPerson()
print( "I am at: ", Person.GetPosition( me ) )
```

See Also

- LocalPlayer.GetPerson

Person.GetRigComponentObjectId

Brief

Returns the specified component type's object ID.

Definition

```
string Person.GetRigComponentObjectId( Person person, string componentType )
```

Arguments

person - The person to query.componentType - The component type to search for. It is case sensitive.

Return Values

The component's object ID or nil if not found.

Description

Returns the specified component type's object ID. The component must be loaded to identity its type.

Examples

```
me = LocalPlayer.GetPerson()
types = Person.GetRigComponentTypes( person )
print( 'type=' .. types[ 1 ] .. ' object id=' .. Person.GetRigComponentObjectId( person, types[ 1 ] )
)
```

See Also

- Person.GetRigComponentTypes

Person.GetRigComponentTypes

Brief

Returns a table of strings for the current rig's component types.

Definition

```
string[] Person.GetRigComponentTypes( Person person )
```

Arguments

person - The person to query.

Return Values

Table of strings for the rig component types.

Description

Returns a table of strings for the current rig's component types. The rig must be loaded to obtain the information.

Examples

```
me = LocalPlayer.GetPerson()
rigCompTypes = Person.GetRigComponentTypes( me )
if rigCompTypes ~= nil then
    for index, item in ipairs( rigCompTypes ) do
        print( 'type = ' .. item )
    end
end
```

See Also

- Person.GetRigComponentObjectId

Person.GetRigField

Brief

Returns the value of the field and nil if not found

Definition

```
string|number|boolean Person.GetRigField( Person person, string field )
```

Arguments

person - The person to query.field - The field to retrieve. Can be one of the following:
"sex" (returns the string "male" or "female")

Return Values

The value of the field and nil if not found. Supported types are boolean, string and number.

Description

Returns the value of the field and nil if not found. The rig must be loaded to get the correct result.

Examples

```
me = LocalPlayer.GetPerson()
print( "My gender is " .. Person.GetRigField( me, "sex" ) )
```

See Also

- LocalPlayer.GetPerson

Person.GetRotationY

Brief

Returns the y-axis rotation of the person.

Definition

```
number Person.GetRotationY( Person person )
```

Arguments

person - The person to query.

Return Values

The rotation in the y-axis of the person in degrees.

Description

Returns the rotation in the y-axis of the specified person in degrees.

Examples

```
me = LocalPlayer.GetPerson()
print( "My rotation is: " .. Person.GetRotationY( me ) )
```

See Also

- LocalPlayer.GetPerson

Person.GetUser

Brief

Returns the User object for this person.

Definition

```
User Person.GetUser( Person person )
```

Arguments

person - The person to query.

Return Values

The User object for the specified person.

Description

Returns the User object attached to the specified person.

Examples

```
local user = person:GetUser()
print(person:GetName() .. (user:IsFriend() and " is " or " is not ") .. "your friend!")
```

See Also

Person.GetVelocity

Brief

Returns the current velocity of the person.

Definition

```
Vector4 GetVelocity( Person person )
locity( Person person, Vector4 outVec )
```

Arguments

person - the person to queryoutVec - The Vector4 to store the result in

Return Values

Returns the current velocity of the person specified if no outVec is supplied.

Description

Returns a vector containing the velocity of the person specified.

Examples

```
local me = LocalPlayer.GetPerson()
elocity is: ", Person.GetVelocity( me ) )
```

See Also

None

Person.GetWidth

Brief

Retrieves the width factor of an avatar.

Definition

```
number Person.GetWidth(Person person)
```

Arguments

person - the person to query.

Return Values

The width factor of the person in the range [-1, 1], or nil if the avatar isn't yet loaded and available.

Description

Return the width factor of the avatar, which is in the range -1 to 1, with -1 being thin, 0 being average, and 1 being fat.

Examples

```
if ( Person.GetWidth() == 1) then BreakChair() end
```

See Also

None

Person.GetWorldBounds

Brief

Retrieve the world-space bounds of a person.

Definition

```
boolean Person.GetWorldBounds(Person person, Vector4 min, Vector4 max)
```

Arguments

person - The person to query.min (out) - Vector4 value to receive the minimum extent.max (out) - Vector4 value to receive the maximum extent.

Return Values

True if the bounds were retrieved successfully, false if the person is not yet loaded.

Description

Get the world-space bounds of a person's model. This will reflect the user's clothing choices and height selection. If the person's avatar is not loaded, this function will return false.

Examples

```
local min = Vector4.Create()
local max = Vector4.Create()
if (LocalPlayer.GetPerson():GetWorldBounds(min, max)) then
    print("Local player avatar is", max:Y() - min:Y(), "metres tall")
end
```

See Also

- Person.GetLocalBounds

Person.HasLock

Brief

true if the person is locked by the calling Lua environment and false otherwise.

Definition

```
boolean Person.HasLock( Person person )
boolean Person.HasLock( Person person , number lockFeatures, boolean any = false )
```

Arguments

person - the person to query.lockFeatures - specific lock features to test to see if they are locked by this lua environmentany - if lockFeatures is specified, then this parameter is used to indicate that if any of the features specified are locked, then return true
otherwise will only return true if all the features specified are locked.

Return Values

true if the person is locked by the calling Lua environment and false otherwise.

Description

true if the person is locked by the calling Lua environment and false otherwise.

Examples

```
if Person.HasLock( person ) == true then
    Person.DoAction( person, "Clap" )
end
```

See Also

- Person.Lock
- Person.Unlock
- Person.IsLocked
- Person.DoAction
- Person.SetBehavior

Person.InterpolatePosition

Brief

Allows the client to interpolate the position of the person in the frames between successive calls to InterpolatePosition.

Definition

```
Person.InterpolatePosition( Person person, Vector4 position, Vector4 velocity, number facingAngle,
number timeOffset = 0)
```

Arguments

person - the person to interpolate the position onposition - the position to use for the interpolationvelocity - the velocity to use for the interpolationfacingAngle - the angle the person is facingtimeOffset - the difference in time between the session clocks (in seconds) - if omitted then defaults to 0.0

Return Values

None.

Description

The calling lua environment must have atleast a Transform lock on the person for the call to succeed This call will only work on NPC persons (will throw a Lua error for the the local player person).

Examples

```
person:InterpolatePosition( recvdPos, recvdVel, recvdAngle, timeOffset )
```

See Also

- Person.Lock
- Person.GetPosition
- Person.GetVelocity

Person.IsCollision

Brief

Reports whether the person's collision is enabled.

Definition

```
boolean Person.IsCollision( Person person )
```

Arguments

person - the person to query.

Return Values

true if collision for the player is on and false otherwise.

Description

Reports whether the person's collision is enabled.

Examples

```
if Person.IsCollision( person ) then
    Person.EnableCollision( person, false )
end
```

See Also

- Person.EnableCollision

Person.IsLoaded

Brief

Indicates when a person's avatar is fully loaded and visible.

Definition

```
boolean Person.IsLoaded(Person person)
```

Arguments

person - The person to query.

Return Values

True if the person's avatar is fully loaded and ready, or false if still loading or invalid.

Description

When a person initially enters the scene, their avatar and associated components may still be loading in the background. Some functionality such as locking is not available until the avatar is ready, so this function can be used to query the loading state of the person. If the person object is invalid, this function will return false.

Examples

```
if (person:IsValid() and person:IsLoaded()) then
    if (not person:Lock(PersonLockType.All)) then
        print("Unable to get lock!")
    end
end
```

See Also

- Person.IsValid

Person.IsLocked

Brief

true if the person is locked by any lua environment and false otherwise.

Definition

```
boolean Person.IsLocked(Person person)
```

Arguments

person - the person to query.

Return Values

true if the person is locked by any lua environment and false otherwise.

Description

true if the person is locked by any lua environment and false otherwise.

Examples

```
if Person.IsLocked( person ) == true and Person.HasLock( person ) == false then
    print( 'Waiting to get the lock' )
end
```

See Also

- Person.Lock
- Person.Unlock
- Person.HasLock
- Person.DoAction
- Person.SetBehavior

Person.IsOnGround

Brief

Return whether the person is on the ground or not.

Definition

```
boolean Person.IsOnGround( Person person )
```

Arguments

person - The person to query

Return Values

true if on the ground, false otherwise.

Description

Return whether the person is on the ground or not.

Examples

```
print( Person.IsOnGround( person ) );
```

See Also

- Person.GetGroundUserId

Person.IsRigUsingComponent

Brief

Returns true if the specified object is being used and false otherwise.

Definition

```
boolean Person.IsRigUsingComponent( Person person, string objectId )
```

Arguments

person - The person to query.objectId - The object ID to test for its usage.

Return Values

true if the specified object is being used and false otherwise.

Description

Returns true if the specified object is being used and false otherwise. Does not require the specified object have have completed its loading for the function to succeed.

Examples

```
me = LocalPlayer.GetPerson()
smartJacketId = "853B950D-ACF84EE7-920D819B-5A9A6C16"
if Person.IsRigUsingComponent( me, smartJacketId ) then
    accessGranted = true
end
```

See Also

- LocalPlayer.GetPerson

Person.IsValid

Brief

Returns whether the handle to the person is valid or not.

Definition

```
boolean Person.IsValid( Person person )
```

Arguments

person - The person to query.

Return Values

Returns whether the handle to the person is valid or not.

Description

Returns whether the handle to the person is valid or not. Invalid persons can be caused if the person leaves the scene while you still have a handle to that person.

Examples

```
if Person.IsValid( randomPerson ) == false then
    person = GetAnotherPerson()
end
```

See Also

- Person.FindInInstance
- Person.IsLoaded

Person.IsVisible

Brief

Return whether the person is visible or not to the user.

Definition

```
boolean Person.IsVisible( Person person )
```

Arguments

None

Return Values

Return whether the person is visible or not

Description

If a user can see a Person in game this will return true, otherwise false. No lock is required since you are only querying the value.

Examples

```
Person.SetVisibile( person, true );
print( Person.IsVisible( person ) );
```

See Also

- Person.Lock

- Person.Unlock

Person.Lock

Brief

Locks a combination of the person's animation, collision state, transform, label or chatbubbles so it can be controlled by script rather than network messages or pad input.

Definition

```
boolean Person.Lock( Person person, number lockType = nil )
```

Arguments

person - the person to query.lockType - Can be one of the following or nil:
PersonLockType.Animation
PersonLockType.Transform
PersonLockType.Collision
PersonLockType.Visibility
PersonLockType.Label
PersonLockType.ChatBubbles
PersonLockType.EmoteMenu
PersonLockType.Wardrobe
PersonLockType.Default = PersonLockType.Animation + PersonLockType.Transform + PersonLockType.Collision + PersonLockType.Visibility
PersonLockType.All - all the lock types together
If lockType is nil then PersonLockType.Default is used.

Return Values

True if the Lua environment was able to obtain the lock (or already has the lock) on the person, otherwise false.

Description

A Lua error is generated if the lockType passed in aren't one of the possible values above. Lua errors are also generated if PersonLockType.Label or PersonLockType.ChatBubbles is used when not in a minigame which supports it. A lock can only be obtained for people currently joined to the active game, and will be released automatically for all people when the local player quits the game. A script should not rely on this automatic behaviour however, and locks should be explicitly released as soon as they are no longer required. *NB To prevent a delay in character loading causing scripts to fail, always ensure that calls to Person.Lock() return true before using any functions which require a locked character. If the function returns false, wait for a short period and then attempt to lock the character again. This can be performed in a co-routine if necessary. 1.55 Added PersonLockType.Wardrobe. This lock type can only be used on the Local Player person, otherwise a Lua error is thrown.

Examples

```
-- Example 1, locking specific types.
Person.Lock( person, PersonLockType.Transform ) -- Locked the transform
Person.Lock( person, PersonLockType.Collision ) -- Unlocked the transform and now locked the collision
Person.Lock( person, PersonLockType.Collision + PersonLockType.Visibility ) -- Now additionally locked
the visibility

-- Example 2, locking transform, collision, animation and visibility.
if Person.Lock( person ) then
    Person.DoAction( person, "clap" )
end
```

See Also

- Person.Unlock
- Person.IsLocked
- Person.HasLock

- Person.DoAction
- Person.SetBehavior

Person.ResetCurrentStatusText

Brief

Reset the status text to show in the over-head person label.

Definition

```
Person.ResetCurrentStatusText( Person person )
```

Arguments

person - The person to update.

Return Values

None.

Description

Reset the status text to show in the over-head person label.

Examples

```
me = LocalPlayer.GetPerson()
Person.ResetCurrentStatusText( me )
```

See Also

- LocalPlayer.GetPerson

Person.SetAnimReg

Brief

Sets the value of the specified animation register for the specified person.

Definition

```
Person.SetAnimReg(Person person, string registerName, number|boolean|string value)
```

Arguments

person - person to set the animation register for.registerName - the name of the animation register to set.value - the value to set the register (the type should agree with the register type).

Return Values

None.

Description

If the register doesn't exist or does not have Lua write access, or the value type and the register type don't match, then an error will be thrown.

Examples

```
person:SetAnimReg("myReg", 0.3)
```

See Also

- Person.GetAnimReg

Person.SetBehavior

Brief

Sets the current behavior of the person

Definition

```
Person.SetBehavior( Person person, string behaviorName, number unlockPeriodSeconds = 0, boolean
snapAnimation = false )
Person.SetBehavior( Person person, string behaviorName, number unlockPeriodSeconds, boolean
snapAnimation, number unlockFeatures )
```

Arguments

person - The Person to set the behavior onbehaviorName - The name of the behavior.unlockPeriodSeconds - if great than 0 then the animation lock on the person is automatically released after
the period specified has elapsed. Defaults to 0.snapAnimation - if true, then removes blend from previous action/behavior (defaults false)unlockFeatures - only used if the unlockPeriodSeconds parameter is greater than 0. If omitted then defaults to releasing the locks that were set when
Person.SetBehavior was called, otherwise will only release the locks specified.

Return Values

Error code. BehaviorError.None if everything was fine, BehaviorError.Invalid if the behavior cannot be set, BehaviorError.NotReady if the person is not ready for setting the behavior

Description

The following behaviors Home behaviors are available for use:
sit_low
sit_mid
sit_high
sit_low_recline
sit_mid_recline
standing
Additionally, any behaviors defined by the developer within an objects repertoire are available for use by the developer. The animation must be locked by the calling lua environment otherwise a lua error is generated. If the behavior specified isn't recognised, then a lua error is thrown.

Examples

```
if Person.HasLock( person ) == true then
    Person.SetBehavior( person, "sit_mid" )
end
```

See Also

- Person.Lock
- Person.Unlock
- Person.IsLocked
- Person.HasLock

- Person.DoAction

Person.SetCurrentStatusText

Brief

Set the status text to show in the over-head person label.

Definition

```
Person.SetCurrentStatusText( Person person, string text, bool persist = false )
```

Arguments

person - The person to update.text - The status text to show in the over-head person label.persist - boolean which determines whether the text is displayed in picture captures.

Return Values

None.

Description

Set the status text to show in the over-head person label. The status text is used to indicate an activity that the user is currently partaking in. This function must not be used without first obtaining permission from your RHQ.

Examples

```
me = LocalPlayer.GetPerson()
Person.SetCurrentStatusText( me, "Racing!" )
```

See Also

None

Person.SetLabelText

Brief

Change the current text label displayed atop the avatar.

Definition

```
Person.SetLabelText( Person person, enumeration setting )
```

Arguments

setting - the current setting to use for player labels.
PersonLabelText.On
PersonLabelText.Off
PersonLabelText.Name

Return Values

None.

Description

With this function, a script can set the labels above a person to show full labels, just the name, or even switch them off. Please note: In order to use this function, the is_allowed_label_change property in the object's mini_game component must be set to true.

Examples

```
local person = LocalPlayer.GetPerson( );
if ( Person.Lock( person, PersonLockType.Label ) == true ) then
    Person.SetLabelText( person, PersonLabelText.Off );
end
```

See Also

- Person.Lock
- Person.Unlock

Person.SetPosition

Brief

Sets the position of the the person, assuming the calling Lua environment owns the lock (with transformation attribute)

Definition

```
SetPosition( Person person, Vector4 position )
```

Arguments

position - the desired world position of the person

Return Values

None.

Description

The transform must be locked by the calling lua environment otherwise a lua error is generated.

Examples

```
Person.SetPosition( person, pos )
```

See Also

- Person.Lock
- Person.Unlock

Person.SetRotationY

Brief

Sets the rotation in the (world) Y axis of the person.

Definition

```
Person.SetRotationY( Person person, number angle)
```

Arguments

person - The Person to set.angle - The world angle in degrees.

Return Values

None.

Description

The transform must be locked by the calling lua environment otherwise a lua error is generated.

Examples

```
Person.SetRotationY( person, 180 )
```

See Also

- Person.Lock
- Person.Unlock

Person.SetVisible

Brief

Sets the visibility of the the person.

Definition

```
Person.SetVisible( Person person, boolean enable )
```

Arguments

enable - whether to switch visibility on or off for that person

Return Values

None.

Description

Sets the visibility of the the person. The visibility must be locked with visibility attribute PersonLockType.Visibility by the calling lua environment otherwise a lua error is generated. If the person is not valid the call is ignored.

Examples

```
Person.SetVisible( person, false )
```

See Also

- Person.Lock
- Person.Unlock
- Person.IsLocked
- Person.HasLock

Person.Unlock

Brief

Unlocks the features specified on the person. If all features are unlocked, then lock is released, allowing other Lua environments to get a lock.

Definition

Arguments

person - the Person to unlock.unlockFlags - the features to unlock. If 0 is passed in will return false.

Return Values

A true return value only indicates that the features requested to be unlocked were unlocked, which may mean that other features are still locked and that the person is not yet free to be locked by another calling environment - Use if (Person.HasLock(person, PersonLockType.All)==false) to verify if lock has been released on the person. *Return value behaviour has changed for 1.45 - see history for previous behaviour.

Description

If all features are unlocked, it then allows other lua environments to lock the person. The features are:
PersonLockType.Animation
PersonLockType.Transform
PersonLockType.Collision
PersonLockType.Visibility
PersonLockType.Label
PersonLockType.ChatBubbles
PersonLockType.EmoteMenu
PersonLockType.Default = PersonLockType.Animation + PersonLockType.Transform + PersonLockType.Collision + PersonLockType.Visibility
PersonLockType.All - all the lock types together

Examples

```
Person.Unlock( person , PersonLockType.Animation + PersonLockType.Transform )
```

See Also

- Person.Lock
- Person.IsLocked
- Person.HasLock
- Person.DoAction
- Person.SetBehavior

Phys3dWorld

Phys3dWorld.Create

Brief

Create a 3d physics world.

Definition

```
Phys3dWorld Phys3dWorld.Create( )
```

Arguments

None

Return Values

The world.

Description

Create a 3D physics world. By default, all rigid bodies in PS Home are loaded into the default physics world, shared by all loaded objects in the current scene. The dynamics engine itself is deterministic, but only if all steps involved in setting up the world and the rigid bodies within it are performed in the same order. As a result, in order for a script to implement a deterministic game involving dynamics it cannot rely solely on the default physics world, as this is being modified by other parties. This function is provided to enable a script to create a "private" physics world for use only by the current script, and thus implement a deterministic game if desired. Scripts should only use this functionality for this purpose, there is no other advantage to creating a physics world manually. Note that you must explicitly add new collision entities to the private world, by default all entities will still be added to the default world.

Examples

```
world = Phys3dWorld.Create()
entity = Entity.Create()
entity:SetCollision("collision", world)
```

See Also

- Entity.SetCollision

Phys3dWorld.Reset

Brief

This resets the world.

Definition

```
Phys3dWorld.Reset( Phys3dWorld world )
```

Arguments

world - The world.

Return Values

None.

Description

This in effect completely clears the physics world. Items would need to be re-added to the world. This is available to developers so they do not have to increase the number of Lua handles.

Examples

```
world = Phys3dWorld.Create( )
Phys3dWorld.Reset( world )
```

See Also

- Phys3dWorld.Create

Phys3dWorld.SetEnabled

Brief

This sets the world to enabled or disabled.

Definition

```
Phys3dWorld.SetEnabled( Phys3dWorld world, boolean enable )
```

Arguments

world - The world.enable - True to enable the world, false otherwise.

Return Values

None.

Description

A world is by default disabled, so items can be added and/or modified at will. When we want the simulation to run, we set the world to be enabled. In effect, items are paused when added to this world until it is enabled.

Examples

```
world = Phys3dWorld.Create( )
Phys3dWorld.SetEnabled( world, true )
```

See Also

- Phys3dWorld.Create

Quaternion

Quaternion.Add

Brief

Adds one quaternion to another, element by element.

Definition

```
Quaternion.Add( Quaternion destination, Quaternion source1, Quaternion source2 )
```

Arguments

destination - Quaternion to store the result of the addition.source1 - Quaternion to be added to source2.source2 - Quaternion to be added to source1.

Return Values

None

Description

Adds source1 and source2 element-wise, storing the result in destination.

Examples

```
result = Quaternion.Create()
q1 = Quaternion.Create( )
q2 = Quaternion.Create( )
Quaternion.Add( result, q1, q2 )        -- result = q1 + q2
```

See Also

- Quaternion.Create

Quaternion.Conjugate

Brief

Conjugate a quaternion.

Definition

```
Quaternion.Conjugate( Quaternion q )
```

Arguments

q - Quaternion to be conjugated.

Return Values

None

Description

Conjugate a quaternion.

Examples

```
q = Quaternion.Create()
Quaternion.Conjugate( q )
```

See Also

- Quaternion.Create

Quaternion.Copy

Brief

Copies the values of one quaternion into another.

Definition

```
Quaternion.Copy( Quaternion destination, Quaternion source )
```

Arguments

destination - The quaternion to copy to.source - The quaternion to be copied.

Return Values

None

Description

Copies the values of the source quaternion and stores them in the destination. Lua has reference semantics for user types, so using q1 = q2 will not copy values but rather make q1 reference q2.

Examples

```
q1 = Quaternion.Create()
q2 = Quaternion.Create()
Quaternion.Copy( q2, q1 )
```

See Also

- Quaternion.Create

Quaternion.Create

Brief

Creates a new Quaternion object using the specified values.

Definition

```
Quaternion Quaternion.Create( )
Quaternion Quaternion.Create( Quaternion source )
Quaternion Quaternion.Create( number x, number y, number z, number w )
Quaternion Quaternion.Create( Matrix44 rotationMatrix )
Quaternion Quaternion.Create( Vector4 axis, number angle )
```

Arguments

source - Quaternion whose values are used to initialize newly created quaternion.x - Number to be written into the first element of the quaternion.y - Number to be written into the second element of the quaternion.z - Number to be written into the third element of the quaternion.w - Number to be written into the fourth element of the quaternion.rotationMatrix - A rotation matrix to be converted to a quaternion.axis - Rotation axis.angle - Rotation angle about the rotation axis in degrees.

Return Values

A new Quaternion instance.

Description

Creates a new Quaternion object using the specified values.

Examples

```
-- Create a default identity (0, 0, 0, 1) quaternion
q1 = Quaternion.Create()

-- Create a quaternion with supplied values
q2 = Quaternion.Create( 0, 0, 0, 1 )

-- Create a quaternion using the values of another
q3 = Quaternion.Create( q2 )

-- Create a quaternion from a rotation matrix
Matrix44 rotation = Matrix44.Create()
q4 = Quaternion.Create( rotation )

-- Create a quaternion from a rotation axis and rotation angle
Vector4 axis = Vector4.Create( 0, 1, 0 )
angle = 45
q5 = Quaternion.Create( axis, angle )
```

See Also

None

Quaternion.Divide

Brief

Performs scalar division on a quaternion.

Definition

```
Quaternion.Divide( Quaternion destination, Quaternion source, number scalar )
```

Arguments

destination - Quaternion to store the result of the division.source1 - Quaternion to divide.scalar - The scalar to divide by.

Return Values

None

Description

Multiplies source by 1/scalar, storing the result in destination.

Examples

```
result = Quaternion.Create()
q1 = Quaternion.Create()
Quaternion.Divide( result, q1, 2 )
```

See Also

- Quaternion.Create

Quaternion.Dot

Brief

Performs a dot product between two quaternions.

Definition

```
number Quaternion.Dot( Quaternion src1, Quaternion src2 )
```

Arguments

src1 - First quaternion to take part in the dot product.src2 - Second quaternion to take part in the dot product.

Return Values

The dot product of the two supplied quaternions.

Description

Performs a dot product between two quaternions.

Examples

```
q1 = Quaternion.Create()
q2 = Quaternion.Create()
result = Quaternion.Dot2( q1, q2 )
```

See Also

None

Quaternion.Length

Brief

Calculates the magnitude of the specified quaternion.

Definition

```
number Quaternion.Length( Quaternion quaternion )
```

Arguments

quaternion - Source quaternion for the calculation.

Return Values

A number holding the magnitude of the specified quaternion.

Description

Calculates the magnitude of the specified quaternion.

Examples

```
q1 = Quaternion.Create()
print( "Quaternion magnitude is: " ..  Quaternion.Length( q1 ) )
```

See Also

None

Quaternion.Multiply

Brief

Multiplies two quaternion by one another or a quaternion by a scalar.

Definition

```
Quaternion.Multiply( Quaternion destination, Quaternion source1, Quaternion source2 )
Quaternion.Multiply( Quaternion destination, Quaternion source, number scalar )
Quaternion.Multiply( Quaternion destination, number scalar, Quaternion source )
```

Arguments

destination - Quaternion to store the result of the multiplication.source1 - The first quaternion argument for the multiplication.source2 - The second quaternion argument for the multiplication.scalar - Number to multiply quaternion components by.

Return Values

None

Description

Multiplies source1 and source2 or source and scalar, storing the result in destination.

Examples

```
result = Quaternion.Create()
v1 = Quaternion.Create()
v2 = Quaternion.Create()
Quaternion.Multiply( result, v1, v2 )
Quaternion.Multiply( result, result, 0.5 )
```

See Also

- Quaternion.Create

Quaternion.Negate

Brief

Negate the components of a quaternion.

Definition

```
Quaternion.Negate( Quaternion q )
```

Arguments

q - Quaternion to be negated.

Return Values

None

Description

Negate the components of a quaternion.

Examples

```
q = Quaternion.Create()
Quaternion.Negate( q )
```

See Also

- Quaternion.Create

Quaternion.Nlerp

Brief

Performs a normalized linear interpolation between two quaternions.

Definition

```
Quaternion.Nlerp( Quaternion destination, Quaternion source1, Quaternion source2, number t )
```

Arguments

destination - Quaternion to store the result of the nlerp.source1 - The first quaternion argument for the nlerp.source2 - The second quaternion argument for the nlerp.t - The interpolation parameter, in range [0, 1].

Return Values

None

Description

Performs a normalized linear interpolation between two quaternions. The shortest path rotation is taken.

Examples

```
result = Quaternion.Create()
q1 = Quaternion.Create( axis, 0 )
q2 = Quaternion.Create( axis, 90 )
Quaternion.Nlerp( result, q1, q2 )
```

See Also

- Quaternion.Create

Quaternion.Normalize

Brief

Normalizes the specified quaternion.

Definition

```
Quaternion.Normalize( Quaternion quaternion )
```

Arguments

quaternion - The quaternion to normalize.

Return Values

None

Description

Normalizes the specified quaternion.

Examples

```
q = Quaternion.Create()
Quaternion.Normalize( q )       -- note that this has no effect to the identity quaternion from Create
```

See Also

- Quaternion.Create

Quaternion.SetAxisAngle

Brief

Sets a quaternion to represent a given axis/angle rotation.

Definition

```
Quaternion.SetAxisAngle( Quaternion destination, Vector4 axis, number angle )
```

Arguments

destination - Quaternion to store the result of the slerp.axis - Rotation axis.angle - Angle of rotation about the rotation axis, in degrees.

Return Values

None

Description

Sets a quaternion to represent a given axis/angle rotation.

Examples

```
q = Quaternion.Create()
axis = Vector4.Create( 0, 1, 0 )
angle = 45
Quaternion.SetAxisAngle( q, axis, angle )
```

See Also

- Quaternion.Create
- Vector4.Create

Quaternion.SetIdentity

Brief

Sets the quaternion to the identity quaternion (0, 0, 0, 1).

Definition

```
Quaternion.SetIdentity( Quaternion quat )
```

Arguments

quat - Quaternion to be set to identity.

Return Values

None

Description

Sets the quaternion to the identity quaternion (0, 0, 0, 1).

Examples

```
q = Quaternion.Create()
Quaternion.SetIdentity( q )     -- this is redundant after using the default constructor however
```

See Also

- Quaternion.Create

Quaternion.SetRotationMatrix

Brief

Sets a quaternion to represent a given rotation matrix.

Definition

```
Quaternion.SetRotationMatrix( Quaternion destination, Matrix44 rotationMatrix )
```

Arguments

destination - Quaternion to store the result of the slerp.rotationMatrix - Rotation matrix.

Return Values

None

Description

Sets a quaternion to represent a given rotation matrix.

Examples

```
q = Quaternion.Create()
m = Matrix44.Create()
Matrix44.SetAxisRotation( m, 0.5, 0.5, 0, 90, MATRIX_REPLACE )
Quaternion.SetRotationMatrix( q, m )
```

See Also

- Matrix44.Create
- Matrix44.SetAxisRotation
- Quaternion.Create

Quaternion.SetW

Brief

Sets the w value of the quaternion.

Definition

```
Quaternion.SetW( Quaternion quaternion, number value )
```

Arguments

quaternion - Quaternion to be set.value - A numerical value that is copied into the w component of the quaternion.

Return Values

None

Description

Sets the w value of the quaternion.

Examples

```
q = Quaternion.Create()
Quaternion.SetW( q, 0 )
```

See Also

- Quaternion.Create
- Quaternion.SetX
- Quaternion.SetY
- Quaternion.SetZ

Quaternion.SetX

Brief

Sets the x value of the quaternion.

Definition

```
Quaternion.SetX( Quaternion quaternion, number value )
```

Arguments

quaternion - Quaternion to be set.value - A numerical value that is copied into the x component of the quaternion.

Return Values

None

Description

Sets the x value of the quaternion.

Examples

```
q = Quaternion.Create()
Quaternion.SetX( q, 0 )
```

See Also

- Quaternion.Create
- Quaternion.SetY
- Quaternion.SetZ
- Quaternion.SetW

Quaternion.SetXyzw

Brief

Sets a Quaternion object with the specified values.

Definition

```
Quaternion.SetXyzw( Quaternion q, number x, number y, number z, number w )
```

Arguments

q - Quaternion to be set.x - Number to be written into the first element of the quaternion.y - Number to be written into the second element of the quaternion.z - Number to be written into the third element of the quaternion.w - Number to be written into the fourth element of the quaternion.

Return Values

None

Description

Sets a Quaternion object with the specified values.

Examples

```
q = Quaternion.Create()
Quaternion.SetXyzw( q, 0, 0, 0, 1 )
```

See Also

- Quaternion.SetX
- Quaternion.SetY
- Quaternion.SetZ
- Quaternion.SetW

Quaternion.SetY

Brief

Sets the y value of the quaternion.

Definition

```
Quaternion.SetY( Quaternion quaternion, number value )
```

Arguments

quaternion - Quaternion to be set.value - A numerical value that is copied into the y component of the quaternion.

Return Values

None

Description

Sets the y value of the quaternion.

Examples

```
q = Quaternion.Create()
Quaternion.SetY( q, 0 )
```

See Also

- Quaternion.Create
- Quaternion.SetX
- Quaternion.SetZ
- Quaternion.SetW

Quaternion.SetZ

Brief

Sets the z value of the quaternion.

Definition

```
Quaternion.SetZ( Quaternion quaternion, number value )
```

Arguments

quaternion - Quaternion to be set.value - A numerical value that is copied into the z component of the quaternion.

Return Values

None

Description

Sets the z value of the quaternion.

Examples

```
q = Quaternion.Create()
Quaternion.SetZ( q, 0 )
```

See Also

- Quaternion.Create
- Quaternion.SetX
- Quaternion.SetY
- Quaternion.SetW

Quaternion.Slerp

Brief

Performs spherical linear interpolation between two quaternions.

Definition

```
Quaternion.Slerp( Quaternion destination, Quaternion source1, Quaternion source2, number t )
```

Arguments

destination - Quaternion to store the result of the slerp.source1 - The first quaternion argument for the slerp.source2 - The second quaternion argument for the slerp.t - The interpolation parameter, in range [0, 1].

Return Values

None

Description

Performs spherical linear interpolation between two quaternions.

Examples

```
result = Quaternion.Create()
q1 = Quaternion.Create( axis, 0 )
q2 = Quaternion.Create( axis, 90 )
Quaternion.Slerp( result, q1, q2, 0.5 )
```

See Also

- Quaternion.Create

Quaternion.Subtract

Brief

Subtracts one quaternion from another, element by element.

Definition

```
Quaternion.Subtract( Quaternion destination, Quaternion source1, Quaternion source2 )
```

Arguments

destination - Quaternion to store the result of the subtraction.source1 - The minuend for the subtraction.source2 - The subtrahend for the subtraction.

Return Values

None

Description

Subtracts source2 from source1 element-wise, storing the result in destination.

Examples

```
result = Quaternion.Create()
q1 = Quaternion.Create()
q2 = Quaternion.Create()
Quaternion.Subtract( result, q1, q2 )       -- result = q1 - q2
```

See Also

- Quaternion.Create

Quaternion.ToAxisAngle

Brief

Get the equivalent rotation in axis/angle representation.

Definition

```
Vector4, number Quaternion.ToAxisAngle( Quaternion q )
```

Arguments

q - Quaternion to be converted to axis angle representation.

Return Values

Vector4 - x,y,z rotation axis, w rotation angle in degrees

Description

Get the equivalent rotation in axis/angle representation.

Examples

```
q = Quaternion.Create()
axisAngle = Quaternion.ToAxisAngle( q )
```

See Also

- Quaternion.Create

Quaternion.W

Brief

Returns the w value of the quaternion.

Definition

```
number Quaternion.W( Quaternion q )
```

Arguments

q - Quaternion to be queried.

Return Values

The fourth element of the quaternion, the w component.

Description

Returns the w value of the quaternion.

Examples

```
q = Quaternion.Create()
print( "Value of W is: " .. Quaternion.W( q ) )
```

See Also

- Quaternion.Create
- Quaternion.X
- Quaternion.Y
- Quaternion.Z

Quaternion.X

Brief

Returns the x value of the quaternion.

Definition

```
number Quaternion.X( Quaternion q )
```

Arguments

q - Quaternion to be queried.

Return Values

The first element of the quaternion, the x component.

Description

Returns the x value of the quaternion.

Examples

```
q = Quaternion.Create()
print( "Value of X is: " .. Quaternion.X( q ) )
```

See Also

- Quaternion.Create
- Quaternion.Y
- Quaternion.Z
- Quaternion.W

Quaternion.Y

Brief

Returns the y value of the quaternion.

Definition

```
number Quaternion.Y( Quaternion q )
```

Arguments

q - Quaternion to be queried.

Return Values

The second element of the quaternion, the y component.

Description

Returns the y value of the quaternion.

Examples

```
q = Quaternion.Create()
print( "Value of Y is: " .. Quaternion.Y( q ) )
```

See Also

- Quaternion.Create
- Quaternion.X
- Quaternion.Z
- Quaternion.W

Quaternion.Z

Brief

Returns the z value of the quaternion.

Definition

```
number Quaternion.Z( Quaternion q )
```

Arguments

q - Quaternion to be queried.

Return Values

The third element of the quaternion, the z component.

Description

Returns the z value of the quaternion.

Examples

```
q = Quaternion.Create()
print( "Value of Z is: " .. Quaternion.Z( q ) )
```

See Also

- Quaternion.Create
- Quaternion.X
- Quaternion.Y
- Quaternion.W

Raycast

Raycast.GetEndPoint

Brief

The value of the end point that was passed in through Raycast.Start.

Definition

```
Vector4 Raycast.GetEndPoint( Raycast raycast )
Raycast.GetEndPoint( Raycast raycast, Vector4 outPoint )
```

Arguments

raycast - The raycast object.outPoint - (out) The vector to receive the end point.

Return Values

The value of the end point that was passed in through Raycast.Start.

Description

Returns the value of the end point that was passed in through Raycast.Start.

Examples

```
thread = coroutine.create( function( ) CastRay( ) end );
function CastRay( )
    ray = Collision.CreateRaycast( );
    rayStart = Vector4.Create( 0, 5, 0 );
    rayEnd = Vector4.Create( 0, -1, 0 );
    Raycast.Start( ray, rayStart, rayEnd, CollisionLayer.Raycast );
    while Raycast.IsReady( ray ) == false do
        coroutine.yield( );
    end

    myPoint = Raycast.GetEndPoint( ray );
    print( tostring( myPoint ) .. " should equal " .. tostring( rayEnd ) );
end

function Update( )
    if coroutine.status( thread ) ~= 'dead' then
        coroutine.resume( thread )
    end
end
```

See Also

- Raycast.Start
- Raycast.GetStartPoint
- Raycast.IsReady

Raycast.GetHitEntity

Brief

Obtain the entity hit by the raycast.

Definition

```
Entity Raycast.GetHitEntity( Raycast raycast )
```

Arguments

raycast - A completed raycast with HasHit() == true

Return Values

The hit entity, or nil if the raycast hit the world or other non-script object.

Description

If a raycast has hit an object, this function will return the entity that was hit if possible, or if the object was not created by script (such as the world, or other non-script-based collision), it will return nil.

Examples

```
thread = coroutine.create( function( ) CastRay( ) end );
function CastRay( )
    ray = Collision.CreateRaycast( );
    rayStart = Vector4.Create( 0, 5, 0 );
    rayEnd = Vector4.Create( 0, -1, 0 );
    Raycast.Start( ray, rayStart, rayEnd, CollisionLayer.Raycast );
    while Raycast.IsReady( ray ) == false do
        coroutine.yield( );
    end

    if Raycast.HasHit( ray ) == true then
        local ent = ray:GetHitEntity();
        if ent ~= nil then
            print( "Entity found at " .. ent:GetPosition( ) )
        end
    end
end

function Update( )
    if coroutine.status( thread ) ~= 'dead' then
        coroutine.resume( thread )
    end
end
```

See Also

- Raycast.Start
- Raycast.HasHit
- Raycast.IsReady

Raycast.GetHitPoint

Brief

Returns the position where the collision occurred if it hit something.

Definition

```
Vector4 Raycast.GetHitPoint( Raycast raycast )
Raycast.GetHitPoint( Raycast raycast, Vector4 outHitPoint )
```

Arguments

raycast - The raycast object.outHitPoint - (out) The vector to receive the hit point.

Return Values

Returns the position where the collision occurred if it hit something.

Description

Returns the position where the collision occurred if it hit something.

Examples

```
function CastRay( )
    ray = Collision.CreateRaycast( );
    rayStart = Vector4.Create( 0, 5, 0 );
    rayEnd = Vector4.Create( 0, -1, 0 );
    Raycast.Start( ray, rayStart, rayEnd, CollisionLayer.Raycast );
    while Raycast.IsReady( ray ) == false do
        coroutine.yield( );
    end
    if Raycast.HasHit( ray ) then
        print( ray:GetHitPoint( ) );
    end
end
```

See Also

- Raycast.Start
- Raycast.HasHit

Raycast.GetReflectedVector

Brief

Returns a normalized vector of the raycast reflected off the surface it hit.

Definition

```
Vector4 Raycast.GetReflectedVector( Raycast raycast )
Raycast.GetEndPoint( Raycast raycast, Vector4 outVector )
```

Arguments

raycast - The raycast object.outVector - (out) The vector to receive the reflected vector.

Return Values

The normalised reflected vector.

Description

Returns a normalised vector of the raycast reflected off the surface it hit. It does not do any further raycasts to see if the reflection collides with anything else. Note: If the hit point is equal to the start point a zero lengh vector is returned.

Examples

```
thread = coroutine.create( function( ) CastRay( ) end );
function CastRay( )
    ray = Collision.CreateRaycast( );
    rayStart = Vector4.Create( 0, 5, 0 );
    rayEnd = Vector4.Create( 0, -1, 0 );
    Raycast.Start( ray, rayStart, rayEnd, CollisionLayer.Raycast );
    while Raycast.IsReady( ray ) == false do
        coroutine.yield( );
    end

    if Raycast.HasHit( ray ) == true then
        mirror = Raycast.GetReflectedVector( ray );
        mirror:Multiply( mirror, rayStart:Length3( ) );
        print( tostring( mirror ) .. " is the mirror image of " .. tostring( rayStart ) .. "
(reflected off a surface normal " .. tostring( ray:GetSurfaceNormal( ) ) .. ")" );
    end
end

function Update( )
    if coroutine.status( thread ) ~= 'dead' then
        coroutine.resume( thread )
    end
end
```

See Also

- Raycast.Start
- Raycast.IsReady
- Raycast.HasHit

Raycast.GetStartPoint

Brief

Returns the value of the start point that was passed in through Raycast.Start.

Definition

```
Vector4 Raycast.GetStartPoint( Raycast raycast )
Raycast.GetStartPoint( Raycast raycast, Vector4 outPoint )
```

Arguments

raycast - The raycast object.outPoint - (out) The vector to receive the start point.

Return Values

The value of the start point that was passed in through Raycast.Start.

Description

Returns the value of the start point that was passed in through Raycast.Start.

Examples

```
thread = coroutine.create( function( ) CastRay( ) end );
function CastRay( )
    ray = Collision.CreateRaycast( );
    rayStart = Vector4.Create( 0, 5, 0 );
    rayEnd = Vector4.Create( 0, -1, 0 );
    Raycast.Start( ray, rayStart, rayEnd, CollisionLayer.Raycast );
    while Raycast.IsReady( ray ) == false do
        coroutine.yield( );
    end

    myPoint = Raycast.GetStartPoint( ray );
    print( tostring( myPoint ) .. " should equal " .. tostring( rayStart ) );
end

function Update( )
    if coroutine.status( thread ) ~= 'dead' then
        coroutine.resume( thread )
    end
end
```

See Also

- Raycast.Start
- Raycast.IsReady

Raycast.GetSurfaceNormal

Brief

Returns the normal of the surface that the raycast hit.

Definition

```
Vector4 Raycast.GetSurfaceNormal( Raycast raycast )
Raycast.GetEndPoint( Raycast raycast, Vector4 outNormal )
```

Arguments

raycast - The raycast object.outNormal - (out) The vector to receive the normal.

Return Values

The normal of the surface that the raycast hit.

Description

Returns the normal of the surface that the raycast hit.

Examples

```
thread = coroutine.create( function( ) CastRay( ) end );
function CastRay( )
    ray = Collision.CreateRaycast( );
    rayStart = Vector4.Create( 0, 5, 0 );
    rayEnd = Vector4.Create( 0, -1, 0 );
    Raycast.Start( ray, rayStart, rayEnd, CollisionLayer.Raycast );
    while Raycast.IsReady( ray ) == false do
        coroutine.yield( );
    end

    direction = rayStart - rayEnd;
    Vector4.Normalize3( direction );
    if Raycast.HasHit( ray ) == true and Vector4.Dot3( Raycast.GetSurfaceNormal( ray ), direction ) >=
0.99 then
        print("We hit the floor (surface normal is almost directly upwards)" );
    end
end

function Update( )
    if coroutine.status( thread ) ~= 'dead' then
        coroutine.resume( thread )
    end
end
```

See Also

- Raycast.Start
- Raycast.IsReady
- Raycast.HasHit

Raycast.GetSurfaceUserId

Brief

Retrieve the user ID of the collision geometry hit by the raycast.

Definition

```
number Raycast.GetSurfaceUserId( Raycast raycast )
```

Arguments

raycast - the raycast to query

Return Values

The user ID, or nil if none is available.

Description

If the raycast has hit, then this function will return the user ID of the collision surface that was hit. If no user ID is available, nil is returned.

Examples

```
thread = coroutine.create( function( ) CastRay( ) end );
function CastRay( )
    ray = Collision.CreateRaycast( );
    rayStart = Vector4.Create( 0, 5, 0 );
    rayEnd = Vector4.Create( 0, -1, 0 );
    Raycast.Start( ray, rayStart, rayEnd, CollisionLayer.Raycast );
    while Raycast.IsReady( ray ) == false do
        coroutine.yield( );
    end

    if Raycast.HasHit( ray ) == true then
        local userID = ray:GetSurfaceUserId( );
        print( "We hit a surface with ID: " .. tostring( userID ) );
    end
end

function Update( )
    if coroutine.status( thread ) ~= 'dead' then
        coroutine.resume( thread )
    end
end
```

See Also

- Raycast.HasHit
- Raycast.HasHit
- Raycast.IsReady

Raycast.HasHit

Brief

Returns true if the raycast hit something and false otherwise.

Definition

```
boolean Raycast.HasHit( Raycast raycast )
```

Arguments

None

Return Values

true if the raycast hit something and false otherwise.

Description

Returns true if the raycast hit something and false otherwise.

Examples

```
thread = coroutine.create( function( ) CastRay( ) end );
function CastRay( )
    ray = Collision.CreateRaycast( );
    rayStart = Vector4.Create( 0, 5, 0 );
    rayEnd = Vector4.Create( 0, -1, 0 );
    Raycast.Start( ray, rayStart, rayEnd, CollisionLayer.Raycast );
    while Raycast.IsReady( ray ) == false do
        coroutine.yield( );
    end
    if Raycast.HasHit( ray ) then
        print( "Hit!" );
    end
end
function Update( )
    if coroutine.status( thread ) ~= 'dead' then
        coroutine.resume( thread )
    end
end
```

See Also

- Raycast.Start
- Raycast.IsReady

Raycast.IsReady

Brief

Returns true if the raycast has finished and false if it hasn't started or is busy calculating.

Definition

```
boolean Raycast.IsReady( Raycast raycast )
```

Arguments

None

Return Values

true if the raycast has finished and false if it hasn't started or is busy calculating.

Description

Returns true if the raycast has finished and false if it hasn't started or is busy calculating.

Examples

```
thread = coroutine.create( function( ) CastRay( ) end );
function CastRay( )
    ray = Collision.CreateRaycast( );
    rayStart = Vector4.Create( 0, 5, 0 );
    rayEnd = Vector4.Create( 0, -1, 0 );
    Raycast.Start( ray, rayStart, rayEnd, CollisionLayer.Raycast );
    while Raycast.IsReady( ray ) == false do
        coroutine.yield( );
    end
end
function Update( )
    if coroutine.status( thread ) ~= 'dead' then
        coroutine.resume( thread )
    end
end
```

See Also

- Collision.CreateRaycast
- Raycast.Start

Raycast.Start

Brief

Starts a basic raycast.

Definition

```
Raycast.Start( Raycast raycast, Vector4 startPoint, Vector4 endPoint, CollisionLayer layer =
CollisionLayer.Raycast)
```

Arguments

startPoint - The start point of the raycast.endPoint - The end point of the raycast.layer - The collision layer of the raycast. Valid layers are:
CollisionLayer.Entity
CollisionLayer.Avatar
CollisionLayer.Camera
CollisionLayer.Raycast
CollisionLayer.User1
CollisionLayer.User2
CollisionLayer.User3
CollisionLayer.User4

Return Values

None.

Description

Starts a basic raycast.

Examples

```
thread = coroutine.create( function( ) CastRay( ) end );
function CastRay( )
    ray = Collision.CreateRaycast( );
    rayStart = Vector4.Create( 0, 5, 0 );
    rayEnd = Vector4.Create( 0, -1, 0 );
    Raycast.Start( ray, rayStart, rayEnd, CollisionLayer.Raycast );
    while Raycast.IsReady( ray ) == false do
        coroutine.yield( );
    end
end
function Update( )
    if coroutine.status( thread ) ~= 'dead' then
        coroutine.resume( thread )
    end
end
```

See Also

- Collision.CreateRaycast
- Raycast.IsReady

ReceivedMessage

ReceivedMessage.GetEventMessage

Brief

Returns an instance of the message that is the focus of the current event.

Definition

```
ReceivedMessage ReceivedMessage.GetEventMessage()
```

Arguments

None

Return Values

The last message received as a ReceivedMessage.

Description

Returns an instance of the message that is the focus of the current event. Only valid during a message received event, as named by the 'on_received_message' property of the 'network' object component in the HDK's Object Editor application.

Examples

```
function OnReceivedMessage()
    local message = ReceivedMessage.GetEventMessage()
    data          = ReceivedMessage.ReadInt32( message )
end
```

See Also

- ReceivedMessage.ReadInt32

ReceivedMessage.GetRemainingSize

Brief

Returns the size of the unread data in the message.

Definition

```
number ReceivedMessage.GetRemainingSize( ReceivedMessage msg )
```

Arguments

msg - The instance of the received message to query.

Return Values

The number of bytes remaining in the message.

Description

Returns the size of the unread data in the message in bytes.

Examples

```
function OnReceivedMessage()
    local message = ReceivedMessage.GetEventMessage()
    bytesToRead   = ReceivedMessage.GetRemainingSize( message )
    print( bytesToRead ) -- Prints total size
    data          = ReceivedMessage.ReadInt32( message )
    bytesToRead   = ReceivedMessage.GetRemainingSize( message )
    print( bytesToRead ) -- Prints total size - 4
end
```

See Also

- ReceivedMessage.GetEventMessage

ReceivedMessage.GetSenderPersonId

Brief

Returns the person id of the sender of the message.

Definition

```
string ReceivedMessage.GetSenderPersonId( ReceivedMessage msg )
```

Arguments

msg - The instance of the received message to query.

Return Values

The sender's person id as a string.

Description

Returns the person id of the sender of the message.

Examples

```
function OnReceivedMessage()
    local message = ReceivedMessage.GetEventMessage()
    personid      = ReceivedMessage.GetSenderPersonId( message )
    print( personid )
end
```

See Also

- Person.FindInInstance
- Person.GetPosition
- ReceivedMessage.GetEventMessage

ReceivedMessage.GetTotalSize

Brief

Returns the total size of a received message.

Definition

```
number ReceivedMessage.GetTotalSize( ReceivedMessage msg )
```

Arguments

msg - The instance of the received message to query.

Return Values

The size in bytes of the message.

Description

Returns the total size of a received message in bytes.

Examples

```
function OnReceivedMessage()
    local message = ReceivedMessage.GetEventMessage()
    messagesize   = ReceivedMessage.GetTotalSize( message )
    print( messagesize )
end
```

See Also

- ReceivedMessage.GetEventMessage

ReceivedMessage.GetType

Brief

Returns the type of a ReceivedMessage.

Definition

```
number ReceivedMessage.GetType( ReceivedMessage msg )
```

Arguments

msg - The instance of the received message to query.

Return Values

The message type as a number.

Description

Returns the user-defined type of a ReceivedMessage. The type is a value that is set by the script when the message is created by the sender.

Examples

```
function OnReceivedMessage()
    local message = ReceivedMessage.GetEventMessage()
    messagetype   = ReceivedMessage.GetType( message )
    print( "Message type is " .. messagetype )
end
```

See Also

- ReceivedMessage.GetEventMessage

ReceivedMessage.ReadBigInt

Brief

Read a BigInt value from a received message.

Definition

```
ReceivedMessage.ReadBigInt( OutboundMessage msg, BigInt bigInt )
```

Arguments

msg - The message.bigInt - The BigInt to receive the value.

Return Values

None.

Description

Reads a BigInt value from a received message. Any size of BigInt is permissible.

Examples

```
function OnReceivedMessage()
    local message = ReceivedMessage.GetEventMessage()
    local bigInt = BigInt.Create(64)
    ReceivedMessage.ReadBigInt( message, bigInt )
    print( "Received big int:", bigInt )
end
```

See Also

- OutboundMessage.WriteBigInt
- BigInt.Create

ReceivedMessage.ReadFloat

Brief

Reads a 32-bit floating point value from the message.

Definition

```
number ReceivedMessage.ReadFloat( ReceivedMessage msg )
```

Arguments

msg - The instance of the received message to read from.

Return Values

32-bit floating point value read from the message packet.

Description

Reads a 32-bit floating point value from the message. When reading past the end of the message buffer, the function will silently fail and return 0.

Examples

```
function OnReceivedMessage()
    local message = ReceivedMessage.GetEventMessage()
    data          = ReceivedMessage.ReadFloat( message )
    print( "Received floating point value is " .. data )
end
```

See Also

- ReceivedMessage.ReadInt8
- ReceivedMessage.ReadInt16
- ReceivedMessage.ReadInt32
- ReceivedMessage.ReadString

ReceivedMessage.ReadInt8

Brief

Reads an 8-bit integer from the message.

Definition

```
number ReceivedMessage.ReadInt8( ReceivedMessage msg )
```

Arguments

msg - The instance of the received message to read from.

Return Values

8-bit integer read from the message packet.

Description

Reads an 8-bit integer (or byte) from the message. When reading past the end of the message buffer, the function will silently fail and return 0.

Examples

```
function OnReceivedMessage()
    local message = ReceivedMessage.GetEventMessage()
    data          = ReceivedMessage.ReadInt8( message )
    print( "Received byte value is " .. data )
end
```

See Also

- ReceivedMessage.ReadInt16
- ReceivedMessage.ReadInt32
- ReceivedMessage.ReadFloat
- ReceivedMessage.ReadString

ReceivedMessage.ReadInt16

Brief

Reads a 16-bit integer from the message.

Definition

```
number ReceivedMessage.ReadInt16( ReceivedMessage msg )
```

Arguments

msg - The instance of the received message to read from.

Return Values

16-bit integer read from the message packet.

Description

Reads a 16-bit integer (or short) from the message. When reading past the end of the message buffer, the function will silently fail and return 0.

Examples

```
function OnReceivedMessage()
    local message = ReceivedMessage.GetEventMessage()
    data          = ReceivedMessage.ReadInt16( message )
    print( "Received short value is " .. data )
end
```

See Also

- ReceivedMessage.ReadInt8

- ReceivedMessage.ReadInt32
- ReceivedMessage.ReadFloat
- ReceivedMessage.ReadString

ReceivedMessage.ReadInt32

Brief

Reads a 32-bit integer from the message.

Definition

```
number ReceivedMessage.ReadInt32( ReceivedMessage msg )
```

Arguments

msg - The instance of the received message to read from.

Return Values

32-bit integer read from the message packet.

Description

Reads a 32-bit integer from the message. When reading past the end of the message buffer, the function will silently fail and return 0.

Examples

```
function OnReceivedMessage()
    local message = ReceivedMessage.GetEventMessage()
    data          = ReceivedMessage.ReadInt32( message )
    print( "Received integer value is " .. data )
end
```

See Also

- ReceivedMessage.ReadInt8
- ReceivedMessage.ReadInt16
- ReceivedMessage.ReadFloat
- ReceivedMessage.ReadString

ReceivedMessage.ReadString

Brief

Reads a string value from the message.

Definition

```
number ReceivedMessage.ReadString( ReceivedMessage msg )
```

Arguments

msg - The instance of the received message to read from.

Return Values

String value read from the message packet.

Description

Reads a string value from the message. When reading past the end of the message buffer, the function will silently fail and return the empty string.

Examples

```
function OnReceivedMessage()
    local message = ReceivedMessage.GetEventMessage()
    data          = ReceivedMessage.ReadString( message )
    print( "Received string value is " .. data )
end
```

See Also

- ReceivedMessage.ReadInt8
- ReceivedMessage.ReadInt16
- ReceivedMessage.ReadInt32
- ReceivedMessage.ReadFloat

Renderer

Renderer.Camera2dGetPosition

Brief

Returns the current overlay's camera position as a Vector4.

Definition

```
Vector4 Renderer.Camera2dGetPosition( Renderer renderer )
Renderer.Camera2dGetPosition( Renderer renderer, Vector4 outPos )
```

Arguments

renderer - The renderer to query.outPos - (out) The vector to receive the position.

Return Values

The current camera's position.

Description

Returns the current overlay's camera position as a Vector4. Only the X and Y components of the returned vector will be valid.

Examples

```
overlayId = Renderer.GetOverlay( renderer )
pos = Renderer.Camera2dGetPosition( renderer )
print( "Camera for overlay " .. overlayId .. " has position " .. pos )
```

See Also

- Renderer.Camera2dSetPosition

Renderer.Camera2dGetRotation

Brief

Returns the current overlay's camera rotation.

Definition

```
number Renderer.Camera2dGetRotation( Renderer renderer )
```

Arguments

renderer - The renderer to query.

Return Values

The camera's rotation in degrees.

Description

Returns the current overlay's camera rotation in degrees.

Examples

```
overlayId = Renderer.GetOverlay( renderer )
rot = Renderer.Camera2dGetRotation( renderer )
print( "Camera for overlay " .. overlayId .. " has rotation " .. rot )
```

See Also

- Renderer.Camera2dSetRotation

Renderer.Camera2dGetScale

Brief

Returns the current overlay's camera scale.

Definition

```
Vector4 Renderer.Camera2dGetScale( Renderer renderer )
Renderer.Camera2dGetScale( Renderer renderer, Vector4 outScale )
```

Arguments

renderer - The renderer to query.outScale - (out) The vector to receive the scale.

Return Values

The camera's scale in the X and Y components of the returned vector.

Description

Returns the current overlay's camera scale. The Z and W components of the returned Vector4 set to 0.

Examples

```
overlayId = Renderer.GetOverlay( renderer )
s = Renderer.Camera2dGetScale( renderer )
print( "Camera for overlay " .. overlayId .. " has scale X=" .. s:X() .. " Y=" .. s:Y() )
```

See Also

- Renderer.Camera2dSetScale

Renderer.Camera2dSetPosition

Brief

Set the current overlay's camera position.

Definition

```
Renderer.Camera2dSetPosition( Renderer renderer, Vector4 position )
```

Arguments

renderer - The renderer to update.position - The new position for the camera.

Return Values

None.

Description

Set the current overlay's camera position to the specified vector. The Z and W components of the specified position are ignored.

Examples

```
Renderer.Camera2dSetPosition( renderer, Vector4.Create( -512, -360 ) )
```

See Also

- Vector4.Create

Renderer.Camera2dSetRotation

Brief

Set the current overlay's camera rotation.

Definition

```
Renderer.Camera2dSetRotation( Renderer renderer, number rotation )
```

Arguments

renderer - The renderer to update.rotation - The rotation in degrees.

Return Values

None.

Description

Set the current overlay's camera rotation in degrees.

Examples

```
Renderer.Camera2dSetRotation( renderer, 180.0 )
```

See Also

None

Renderer.Camera2dSetScale

Brief

Set the current overlay's camera scale.

Definition

```
Renderer.Camera2dSetScale( Renderer renderer, Vector4 scale )
Renderer.Camera2dSetScale( Renderer renderer, number x, number y )
```

Arguments

renderer - The renderer to update.scale - The 2D scale values as the X and Y components of the vector. The Z and W components are ignored.x - X scale value.y - Y scale value.

Return Values

None.

Description

Set the current overlay's camera scale. The scale values are set into a 2D "view" matrix, and thus the scaling of the result will be the reciprocal of the values. For example, setting scale to (0.5, 0.5) will double the size of the output, (2.0, 2.0) will halve it. Setting a scale of 0 will make the overlay invisible. Note that currently text will not be scaled (nor rotated).

Examples

```
Renderer.Camera2dSetScale( renderer, 0.5, 2.0 )
Renderer.Camera2dSetScale( renderer, Vector4.Create( 0.5, 2.0 ) )
```

See Also

- Renderer.Camera2dGetScale

Renderer.Create

Brief

Creates a new Renderer object.

Definition

```
Renderer Renderer.Create( Vector4 pos=[0, 0, 0, 0], Vector4 size=[1280, 720, 0, 0], Vector4 scale=[1,
1, 1, 1], type = 'normal' )
```

Arguments

pos - The position of the renderer relative to where its attached to.size - The size of the renderer relative to where its attached to.scale - The scale of the renderer relative to where its attached to.type - The type of the renderer. Can be one of the following:
"normal" - Automatically hidden when using the Menu Screen or in the Navigator etc.
"global" - Shows always, only hidden when wardrobe is active.
"scene" - Automatically hidden when using the wardrobe or in the Navigator but not hidden when it renders in the scene.

Return Values

A new Renderer instance.

Description

Creates a new Renderer object. If the renderer does not have a target set, it will default to being full screen with a size of 1280x720. If it is targeted at a screen, then the position is relative to that screen. Each renderer consists of 16 overlays and each overlay has its own camera. A use case for overlays can be having the HUD on one overlay, the game on another and the background sprites on a third.

Examples

```
renderer = Renderer.Create()
```

See Also

- Renderer.SetTarget

Renderer.DrawCircle2d

Brief

Draws a 2-dimensional circle.

Definition

```
Renderer.DrawCircle2d( Renderer renderer, Vector4 center, number radius, number segmentCount = 16,
Vector4 rgba = [1,1,1,1] )
Renderer.DrawCircle2d( Renderer renderer, number x, number y, number radius, number segmentCount = 16,
Vector4 rgba = [1,1,1,1] )
```

Arguments

renderer - Target renderer for the draw call.center - The center of the circle.x - The x position of the circle's center.y - The y position of the circle's center.radius - The radius of the circle.segmentCount - The number of segments to use to make the circle.rgba - RGBA values determine the color of the circle.

Return Values

None.

Description

Draws a 2-dimensional circle. The color defaults to white.

Examples

```
-- Render a blue circle
Renderer.DrawCircle2d( renderer, Vector4.Create( 0, 0 ), 60, 10, Vector4.Create( 0, 0, 1, 1 ) );
```

See Also

None

Renderer.DrawLine2d

Brief

Draws a 2-dimensional line.

Definition

```
Renderer.DrawLine2d( Renderer renderer, Vector4 pos1, Vector4 pos2, Vector4 rgba = [1,1,1,1] )
Renderer.DrawLine2d( Renderer renderer, number x1, number y1, number x2, number y2, Vector4 rgba =
[1,1,1,1] )
```

Arguments

renderer - Target renderer for the draw call.pos1 - The first point in the line. The Z and W components of the vector will be ignored.pos2 - The second point in the line. The Z and W components of the vector will be ignored.x1 - The first point's X coordinate.y1 - The first point's Y coordinate.x2 - The second point's X coordinate.y2 - The second point's Y coordinate.rgba - RGBA values determine the color of the line.

Return Values

None.

Description

Draws a 2-dimensional line. The color defaults to white.

Examples

```
-- Render a blue line
Renderer.DrawLine2d( renderer, Vector4.Create( 0, 0 ), Vector4.Create( 100, 200 ), Vector4.Create( 0,
0, 1, 1 ) )
```

See Also

- Renderer.DrawText2d

Renderer.DrawLineList2d

Brief

Draws a list of 2D lines.

Definition

```
Renderer.DrawLineList2d( Renderer renderer, Vector4[] lineList, Vector4 rgba = [ 1, 1, 1, 1 ] )
```

Arguments

renderer - Target renderer for the draw call.lineList - Table of vector pairs defining the 2D line segments.rgba - Color of the line list. Defaults to

white.

Return Values

None.

Description

Draws a list of 2D lines.

Examples

```
lineList = {}
lineList[1] = Vector4.Create( 10, 10 )
lineList[2] = Vector4.Create( 20, 20 )
Renderer.DrawLineList2d( renderer, lineList )
```

See Also

- Vector4.Create

Renderer.DrawQuad2d

Brief

Draw a single quad with specified vertex details.

Definition

```
Renderer.DrawQuad2d(Renderer renderer, table points, table uvs, table colors, Texture texture = nil)
```

Arguments

renderer - Renderer to use.points - An array of 8 numbers, representing 4 XY pairs, the corner vertices of the rect.uvs - An array of 8 numbers, representing 4 UV pairs, the texture cooordinates of the rect corners. Must be nil if no
texture is specified.colors - An array of 4 Vector4 values, representing the color for each vertex. If nil, white (1, 1, 1, 1) is assumed.texture - Texture to use. If nil, no texture is applied and the UV coordinates must be nil.

Return Values

None.

Description

This function is a more flexible alternative to DrawRect2d. All 4 vertices of the quad are specified and the UV (texture) coordinates and colors can be set. Only one texture can be applied to the quad. DrawRect2d will use less memory and be more performant, so prefer that unless you require the additional options this function provides.

Examples

```
points = { -100, 100, 100, 100, 100, -100, -100, -100 }     -- 200x200 square centered on origin
uvs = { 0, 0, 1, 0, 1, 1, 0, 1 }
colors = { Vector4.Create(1, 0, 0, 1), Vector4.Create(1, 0, 0, 1), Vector4.Create(0, 0, 1, 1),
Vector4.Create(0, 0, 1, 1) }
renderer:DrawQuad2d(points, uvs, colors, texture)
```

See Also

- Renderer.DrawRect2d

Renderer.DrawQuads2d

Brief

Draws 2D quads using data contained within MemoryContainers.

Definition

```
Renderer.DrawQuads2d( Renderer renderer, number startVertex, number quadCount, Texture texture=nil,
Matrix44 matrix=nil, MemoryContainer positionData, MemoryContainer uvData=nil, MemoryContainer
colorData|Vector4 colorVec )
```

Arguments

renderer - The renderer to use.quadCount - The number of quads to render.texture - The texture to use or nil if untextured.matrix - The world matrix for the quads or nil if not required.positionData - A MemoryContainer containing positions for the quads. Each quad is made up of 4 vertices and each vertex has an X and Y position which are of type Float32.startVertex - The first vertex index to start from. A startVertex of 0 is the first vertex as MemoryContainer data starts at 0 as well. This applies to all MemoryContainers.uvData - A MemoryContainer containing the texture UVs for the quads. Each quad is made up of 4 vertices and each vertex has a U and V value which are of type Float32.colorData - A MemoryContainer containing the color for the quads. Each quad is made up of 4 vertices and each vertex has 4 color channels (Red, Gree, Blue, Alpha) which are type UInt8 and range from 0 to 255.colorVec - A vector to use for all the quads instead of using colorData.

Return Values

None.

Description

This function lets you draw multiple textured or untextured quads at once using a maximum of one texture with the quad data stored in the supplied memory containers. You can specify the offset of the memory containers and the number of quads you want to draw so you could use the same set of memory containers and draw sections of the data with different matrices or textures. If a matrix is specified then all of the positions are transformed with the matrix. Example usages are custom fonts, especially where there is a large body of static text that you wish to render so one function can set up your memory containers once and then render it each frame with Renderer.DrawQuads2d. Another usage is to have static background data such as a shoot-em up or platform game where the memory containers are set up once and then you only need to do one call per frame to render it all. You can also update the memory containers dynamically for usages such as a 2d particle system or when you are required to renderer hundreds of sprites.

Note: The memory container set functions can take arrays of values to minimize the API calls when setting up your memory containers.

Examples

```
Renderer.DrawQuads2d( g_renderer, 0, g_quadCount, g_fontTexture, g_worldMatrix, g_positions, g_uvs,
g_cols )
```

See Also

- MemoryContainer.Create

Renderer.DrawRect2d

Brief

Draws a solid 2-dimensional rectangle.

Definition

```
Renderer.DrawRect2d( Renderer renderer, Vector4 bottomLeftPos, Vector4 topRightPos, Vector4 rgba =
[1,1,1,1], Texture texture = nil )
Renderer.DrawRect2d( Renderer renderer, number left, number bottom, number right, number top, Vector4
rgba = [1,1,1,1], Texture texture = nil )
```

Arguments

renderer - Target renderer for the draw call.bottomLeftPos - The bottom left position of the rectangle.topRightPos - The top right position of the rectangle.left - The left coordinate of the rectangle.right - The right coordinate of the rectangle.bottom - The bottom coordinate of the rectangle.top - The top coordinate of the rectangle.rgba - The color of the rectangle.texture - The texture to use.

Return Values

None.

Description

Draws a solid 2-dimensional rectangle. The color defaults to white and it is untextured unless a texture is specified.

Examples

```
-- A red rectangle is produced in this example:
renderer = Renderer.Create()
Renderer.DrawRect2d( renderer, Vector4.Create( 0, 0 ), Vector4.Create( 100, 200 ), Vector4.Create( 1,
0, 0, 1 ) );
```

See Also

- Renderer.Create

Renderer.DrawSprite

Brief

Draws a sprite.

Definition

```
Renderer.DrawSprite( Renderer renderer, Sprite sprite )
```

Arguments

renderer - Target renderer for the draw call.sprite - The sprite to draw.

Return Values

None.

Description

Draws a sprite.

Examples

```
renderer = Renderer.Create()
sprite = Sprite.Create()
Sprite.SetTexture( sprite, Texture.Find( "texture.dds" ) )
Renderer.DrawSprite( renderer, sprite )
```

See Also

- Renderer.Create
- Sprite.Create

- Sprite.SetTexture
- Renderer.DrawSprites

Renderer.DrawSprites

Brief

Draws an array of sprites.

Definition

```
Renderer.DrawSprites( Renderer renderer, Sprite[] sprites, number startIndex = 1, number endIndex =
#sprites )
```

Arguments

renderer - Target renderer for the draw call.sprites - A table of sprites to draw.startIndex - The index in the table to start drawing from. Defaults to 1.endIndex - The index in the table to stop drawing to. Defaults to the length of the table.

Return Values

None.

Description

Draws one or more sprites.

Examples

```
renderer = Renderer.Create()
sprites = { Sprite.Create(), Sprite.Create(), Sprite.Create(), Sprite.Create() }
Sprite.SetTexture( sprites[1], Texture.Find( "texture1.dds" ) )
Sprite.SetTexture( sprites[2], Texture.Find( "texture2.dds" ) )
Sprite.SetTexture( sprites[3], Texture.Find( "texture3.dds" ) )
Sprite.SetTexture( sprites[4], Texture.Find( "texture4.dds" ) )
Renderer.DrawSprites( renderer, sprites ) -- Draw all 4 sprites
Renderer.DrawSprites( renderer, sprites, 2, 4 ) -- Draw 2nd, 3rd and 4th sprite.
```

See Also

- Renderer.Create
- Sprite.Create
- Sprite.SetTexture
- Renderer.DrawSprite

Renderer.DrawText2d

Brief

Draws text using the default font.

Definition

```
Renderer.DrawText2d( Renderer renderer, Vector4 position, string text, Vector4 rgba = [1,1,1,1],
number maxWidth = -1 )
Renderer.DrawText2d( Renderer renderer, number x, number y, string text, Vector4 rgba = [1,1,1,1],
number maxWidth = -1 )
```

Arguments

renderer - Target renderer for the draw call.position - The x,y position of the text. The Z and W components of the vector will be ignored.x - The x position of the text.y - The y position of the text.text - The text to render.rgba - The color of the text.maxWidth - The maximum width in pixels the text can be before it goes onto the next line.

Return Values

None.

Description

Draws text using the default font. The color defaults to white.

Examples

```
Renderer.DrawText2d( renderer, Vector4.Create( 5, 12 ), "hello" )
```

See Also

- Vector4.Create

Renderer.DrawTriList2d

Brief

Renders a list of 2D triangles.

Definition

```
Renderer.DrawTriList2d( Renderer renderer, Vector4[] points, Vector4 color = [1,1,1,1] )
```

Arguments

renderer - Target renderer for the draw call.points - The list of corner coordinates for the triangles. Must be multiple of 3.color - The color for the triangles. Defaults to white.

Return Values

None.

Description

Renders a list of 2D triangles. The color for the triangles defaults to white.

Examples

```
vertList = {}
vertList[1] = Vector4.Create( 10, 10 )
vertList[2] = Vector4.Create( 20, 20 )
vertList[3] = Vector4.Create( 10, 20 )
green = Vector4.Create( 0, 1, 0, 1 )
Renderer.DrawTriList2d( renderer, vertList, green )
```

See Also

- Vector4.Create

Renderer.GetOverlay

Brief

Returns the current overlay.

Definition

```
number Renderer.GetOverlay( Renderer renderer )
```

Arguments

renderer - The renderer to query.

Return Values

The initial overlay is 0 and it can range from 0 to 15.

Description

Returns the current overlay.

Examples

```
-- Save the current overlay
previousOverlay = Renderer.GetOverlay( renderer )
Renderer.SetOverlay( renderer, uiOverlay )

-- Render UI

-- Restore old overlay
Renderer.SetOverlay( renderer, previousOverlay )
```

See Also

- Renderer.SetOverlay

Renderer.GetPosition

Brief

Returns the current position of the renderer.

Definition

```
Vector4 Renderer.GetPosition( Renderer renderer )
Renderer.GetPosition( Renderer renderer, Vector4 outPos )
```

Arguments

renderer - The renderer to query.outPos - (out) The vector to receive the position.

Return Values

The current position of the renderer.

Description

Returns the current position of the renderer.

Examples

```
print( "The current position is " .. Renderer.GetPosition( renderer ) )
```

See Also

None

Renderer.GetScale

Brief

Returns the current scale of the renderer.

Definition

```
Vector4 Renderer.GetScale( Renderer renderer )
Renderer.GetScale( Renderer renderer, Vector4 outScale )
```

Arguments

renderer - The renderer to query.outScale - (out) The vector to receive the scale.

Return Values

The current scale of the renderer.

Description

Returns the current scale of the renderer.

Examples

```
print( "The current scale is " .. Renderer.GetScale( renderer ) )
```

See Also

None

Renderer.GetSize

Brief

Returns the current size of the renderer.

Definition

```
Vector4 Renderer.GetSize( Renderer renderer )
Renderer.GetSize( Renderer renderer, Vector4 outSize )
```

Arguments

renderer - The renderer to query.outSize - (out) The vector to receive the size.

Return Values

The current size of the renderer.

Description

Returns the current size of the renderer.

Examples

```
print( "The current size is " .. Renderer.GetSize( renderer ) )
```

See Also

None

Renderer.GetTextSize

Brief

Calculates the pixel width of the text using the current font.

Definition

```
Vector4 Renderer.GetTextSize( Renderer renderer, string text, number maxWidth = -1 )
```

Arguments

renderer - The renderer to query.text - The text to measure.maxWidth - The text to measure.

Return Values

A vector containing the pixel width and height of the text.

Description

Calculates the pixel width of the text using the current font. The size includes the current font's scale. Only the X and Y components of the returned vector are valid, where the X component corresponds to the specified text's width in pixels, and the Y component corresponds to its height. The Z and W components are set to zero and can be ignored. If maxWidth is specified the size is calculated with a maximum width that mirrors the DrawText2d maxWidth parameter.

Examples

```
size = Renderer.GetTextSize( renderer, "Hello World!" )
print( "Text width is " .. Vector4.X( size ) )
print( "Text height is " .. Vector4.Y( size ) )
```

See Also

- Renderer.DrawText2d

Renderer.SetBlendMode2d

Brief

Sets the blend mode of the specified renderer's current overlay.

Definition

```
Renderer.SetBlendMode2d( Renderer renderer, number preset )
```

Arguments

renderer - The renderer to set the blend function on.preset - Blend mode preset to access well known definitions. Can be one of the following:
BlendMode2d.Blend
BlendMode2d.Add
BlendMode2d.Multiply
BlendMode2d.Xor

Return Values

Description

This works on the current overlay. All overlays default to BlendMode2d.Blend every frame. You can Change the blend mode state between render calls, for example changing the blend to additive, drawing several sprites, then changing the blend to multiply and drawing quads with Renderer.DrawQuads2d.

Examples

```
Renderer.SetBlendMode2d( renderer, BlendMode2d.Blend ); --Revert to normal blend mode
Renderer.SetBlendMode2d( renderer, BlendMode2d.Add ); --Set to additive
```

See Also

- Material.SetBlendMode

Renderer.SetFont

Brief

Sets the current font for rendering.

Definition

```
Renderer.SetFont( Renderer renderer, string fontName )
Renderer.SetFont( Renderer renderer, Font font )
```

Arguments

renderer - The renderer to use.fontName - The name of the font to use. Values can be one of the following:
"13reg" - the default for Renderer
"13bold"
"18reg"
"18bold"
"26reg"
"26bold"
"60bold"font - A valid custom Font object.

Return Values

None.

Description

Sets the current font for rendering. The default is "13reg".

Examples

```
local text = 'big font'
Renderer.SetFont( renderer, "60bold" )
local pos = Vector4.Create( 5 + Renderer.GetTextSize( text ) * 0.75, 12 )
Renderer.DrawText2d( renderer, pos, text )
...
font = Font.Create( myFontResource, 8 )     -- or Font.Create( "MyFontResourceName", 8 )
Renderer.SetFont( renderer, font )
Renderer.DrawText2d( renderer, textPos, "Check out my custom font!" )
```

See Also

- Font.Create

Renderer.SetFontHorzAlignment

Brief

Sets the horizontal alignment of the font when rendering text.

Definition

```
Renderer.SetFontHorzAlignment( Renderer renderer, string alignment )
```

Arguments

renderer - The renderer to update.alignment - The horizontal alignment of text to be renderered. Can be one of the following:
"left"
"center"
"right"

Return Values

None.

Description

Sets the horizontal alignment of the font when rendering text. When a renderer is created the default value is "left".

Examples

```
Renderer.SetFontHorzAlignment( renderer, "center" )
```

See Also

- Renderer.SetFontVertAlignment

Renderer.SetFontScale

Brief

Sets the font scale.

Definition

```
Renderer.SetFontScale( Renderer renderer, number scaleX, number scaleY = scaleX )
```

Arguments

renderer - The renderer to update.scaleX - The horizontal scale in the x-axis.scaleY - The vertical scale in the y-axis.

Return Values

None.

Description

Sets the font scale.

Examples

```
Renderer.SetFontScale( renderer, 10 )
```

See Also

- Renderer.DrawText2d

Renderer.SetFontVertAlignment

Brief

Sets the vertical alignment of the font when rendering text.

Definition

```
Renderer.SetFontVertAlignment( Renderer renderer, string alignment )
```

Arguments

renderer - The renderer to update.alignment - The vertical alignment of text to be renderered. Can be one of the following:
"top"
"center"
"bottom"

Return Values

None.

Description

Sets the vertical alignment of the font when rendering text. When a renderer is created the default value is "top".

Examples

```
Renderer.SetFontVertAlignment( renderer, "top" )
```

See Also

- Renderer.SetFontHorzAlignment

Renderer.SetOverlay

Brief

Sets the current overlay.

Definition

```
Renderer.SetOverlay( Renderer renderer, number overlay )
```

Arguments

renderer - The renderer to update.overlay - The overlay id which ranges from 0 to 15.

Return Values

None.

Description

Sets the current overlay. Overlays are drawn in the order 0 to 15.

Examples

```
-- Save the current overlay
previousOverlay = Renderer.GetOverlay( renderer )
Renderer.SetOverlay( renderer, uiOverlay )

-- Render UI

-- Restore old overlay
Renderer.SetOverlay( renderer, previousOverlay )
```

See Also

- Renderer.GetOverlay

Renderer.SetPosition

Brief

Sets the renderer to the specified position.

Definition

```
Renderer.SetPosition( Renderer renderer, Vector4 position )
```

Arguments

renderer - The renderer to update.position - The new position of the renderer.

Return Values

None.

Description

Sets the renderer to the specified position. The default is [0, 0, 0, 1].

Examples

```
renderer = Renderer.Create()
Renderer.SetPosition( renderer, Vector4.Create( -100, 42 ) )
```

See Also

- Renderer.Create
- Vector4.Create

Renderer.SetScale

Brief

Sets the renderer to the specified scale.

Definition

```
Renderer.SetScale( Renderer renderer, Vector4 scale )
```

Arguments

renderer - The renderer to update.scale - The new scale of the renderer.

Return Values

None.

Description

Sets the renderer to the specified scale. The default is [1, 1, 1, 1].

Examples

```
renderer = Renderer.Create()
Renderer.SetScale( renderer, Vector4.Create( 2, 2 ) )
```

See Also

- Renderer.Create
- Vector4.Create

Renderer.SetSize

Brief

Sets the renderer to the specified size.

Definition

```
Renderer.SetSize( Renderer renderer, Vector4 size )
```

Arguments

renderer - The renderer to update.size - The new size of the renderer.

Return Values

None.

Description

Sets the renderer to the specified size. The default is [1280, 720, 0, 0]. Anything outside of the size will be clipped.

Examples

```
renderer = Renderer.Create()
Renderer.SetSize( renderer, Vector4.Create( 1024, 720 ) )
```

See Also

- Renderer.Create
- Vector4.Create

Renderer.SetTarget

Brief

Sets the target of the renderer.

Definition

```
Renderer.SetTarget( Renderer renderer, Screen screen )
```

Arguments

renderer - The renderer to update.screen - The screen to render to.

Return Values

None.

Description

The target of the renderer is set to a screen so instead of rendering in full screen mode it is rendered to the specified screen.

Examples

```
renderer = Renderer.Create()
screen = ArcadeGame.GetScreen()
Renderer.SetTarget( renderer, screen )
```

See Also

- ArcadeGame.GetScreen
- Scene.FindScreen
- Screen.Create

Resource

Resource.CancelRequest

Brief

Cancels a pending resource request.

Definition

```
Resource.CancelRequest( Resource resource )
```

Arguments

resource - The resource to cancel the request for.

Return Values

None.

Description

Cancels a pending resource request.

Examples

```
Resource.CancelRequest( res )
```

See Also

- Resource.Request

Resource.DebugDumpToTty

Brief

Debug function to dump a resource content and/or metadata to TTY.

Definition

```
Resource.DebugDumpToTty(Resource resource)
```

Arguments

resource - The resource to dump.

Return Values

None.

Description

Dumps available information about the resource contents to TTY. File resources will be dumped in their entirety, binary resources will have any available human-readable metadata dumped. Note all resource types are currently supported, presently files/xml and textures are. This function is only available if the debug flag is set on the Lua component, and is not available to live content.

Examples

```
url = "http://www.example.com/data.xml"
xmlData1 = Resource.Request( url, "file" )
...
xmlData1:DebugDumpToTty()
```

See Also

- Resource.DebugSave

Resource.DebugSave

Brief

Debug function to save a resource to the home directory.

Definition

```
boolean Resource.DebugSave(Resource resource, string filename)
```

Arguments

resource - The resource to dump.filename - The filename to use (including extension).

Return Values

True on success or false on failure.

Description

Dumps the contents of a resource to the home directory supplied via Target Manager. The operation is synchronous so be aware that large files may involve a long pause. Currently only file/xml is supported. This function is only available if the debug flag is set on the Lua component, and is not available to live content.

Examples

```
url = "http://www.example.com/data.xml"
xmlData1 = Resource.Request( url, "file" )
...
xmlData1:DebugSave("debugXml.xml")
```

See Also

- Resource.DebugDumpToTty

Resource.Exists

Brief

Check if a specified resource exists in the object or a loaded resource pack.

Definition

```
boolean Resource.Exists(string resourceName, ResourcePack resourcePack = nil)
```

Arguments

resourceName - Name of the resource to search for.resourcePack - A loaded ResourcePack object to search in.

Return Values

True if the resource is present in the object or resource pack, or false otherwise.

Description

Checks for a named resource in the current object or a loaded resource pack.

Examples

```
if (Resource.Exists("gun")) then
    gunEntity:SetModel("gun")
end
```

See Also

- Resource.Find

Resource.Find

Brief

Finds the specified resource.

Definition

```
Resource Resource.Find( string resourceName, ResourcePack pack = nil )
```

Arguments

resourceName - The name of the resource to find.pack - ResourcePack from which to load the resource. Must be fully loaded.

Return Values

The Resource object.

Description

Finds the specified resource. The resource name is the name set for a resource in the HDK's Object Editor application. Alternatively, the resource name can be the name set for a scripted asset in the HDK's Scene Editor application. If the resource is not found an error occurs.

Examples

```
res = Resource.Find( "laserGun" )
print( Resource.GetName( res ) )
```

See Also

None

Resource.GetData

Brief

Depending on the type of resource it returns the corresponding object such as a texture.

Definition

```
variant Resource.GetData( Resource resource )
```

Arguments

resource - The resource to query.

Return Values

Depending on the type of resource it returns the corresponding object.

Description

Depending on the type of resource it returns the corresponding object such as a texture. Supported resource types are:
Texture
SoundBank
SoundStream
Texture
Repertoire

Examples

```
texture = Resource.GetData( textureRes )
textureWidth = Texture.GetWidth( texture )
textureHeight = Texture.GetHeight( texture )
```

See Also

- Resource.Request
- MemoryContainer.Create

Resource.GetDownloadProgress

Brief

Retrieve the download progress of a resource request.

Definition

```
number Resource.GetDownloadProgress( Resource resource )
```

Arguments

resource - A requested resource that is currently loading.

Return Values

A number between 0 and 1 representing the download progress.

Description

If the resource is not downloading 0 will be returned, or if it is fully downloaded 1 is returned. The IsLoading and IsLoaded functions should be preferred if simply checking for the load state. Note that the progress is the download progress only, other tasks may need to be performed after downloading and so the download progress may return 1 while IsLoaded is still false. Only when IsLoaded returns true is the load complete. Also note that a resource may be locally cached, and scripts should not assume that all requests will result in a download.

Examples

```
function OnUpdate()
    if (resource:IsLoading()) then
        local progress = resource:GetDownloadProgress()
        if (progress == 0) then
            print("Requesting...")
        elseif (0 < progress and progress > 1) then
            print("Downloading...")
            RenderProgressBar(progress)
        else
            print("Processing...")
        end
    end
end
```

See Also

- Resource.Request
- Resource.IsLoading
- Resource.IsLoaded

Resource.GetName

Brief

Returns the resource name.

Definition

```
string Resource.GetName( Resource resource )
```

Arguments

resource - The resource to query.

Return Values

The resource name.

Description

Returns the resource name. This function should only be used for resources obtained from a call to Resource.Find.

Examples

```
res = Resource.Find( 'laserGun' )
print( Resource.GetName( res ) )
```

See Also

- Resource.Find

Resource.GetStatusCode

Brief

Retrieve the status code for a requested resource download.

Definition

```
number Resource.GetStatusCode(Resource resource)
```

Arguments

resource - A requested resource that is currently loading.

Return Values

A status code (see below).

Description

For HTTP downloads, the status code will reflect the HTTP response status code, such as 404 in the case of file not found. The status code can be used to determine the cause of failure in the case that IsLoaded() == false and IsLoading() == false, which indicates an error has occurred. If no status code is available 0 is returned. Note that calling GetStatusCode in cases other than load failure is not guaranteed to produce an expected result.

Examples

```
resource = Resource.Request("http://www.example.com/myXmlFile.xml", "xml")
resourceDone = false

function OnUpdate()
    if (resource and not resourceDone) then
        if (resource:IsLoaded()) then
            -- use resource
            resourceDone = true
        elseif (not resource:IsLoading()) then
            -- error case: resource is not loaded but also not loading
            local code = resource:GetStatusCode()
            if (code == 404) then
                print("File has gone missing!")
            else
                print("HTTP download fail:", code)
            end
            resourceDone = true
        else
            print("Progress:", resource:GetDownloadProgress())
        end
    end
end
```

See Also

- Resource.Request
- Resource.GetDownloadProgress

Resource.GetSystemResource

Brief

Get a system resource for use by the script.

Definition

```
Resource Resource.GetSystemResource( SysResources resourceId )
```

Arguments

resourceId - ID of system resource to load (see details).

Return Values

The resource object.

Description

Some resources are loaded by the client and are available to all scripts. These resources will be treated as dynamically loaded (similarly to Resource.Request) but will be shared with the client and not require downloading. The available resource IDs are:

- SysResources.TexPsStoreLogo

Examples

```
resource = Resource.GetSystemResource( SysResources.TexPsStoreLogo )
...
if (resource:IsLoaded()) then
    local tex = resource:GetData()
    Renderer.DrawRect2d( renderer, Vector4.Create( 0, 0 ), Vector4.Create( 200, 50 ), Vector4.Create(
1, 1, 1, 1 ), tex )
end
```

See Also

- Resource.Request
- Resource.Find

Resource.IsLoaded

Brief

Returns true if the resource has been loaded successfully.

Definition

```
boolean Resource.IsLoaded( Resource resource )
```

Arguments

resource - The resource to query.

Return Values

true if the resource has been loaded successfully and false otherwise.

Description

Returns true if the resource has been loaded successfully.

Examples

```
local isLoaded = Resource.IsLoaded( res )
```

See Also

- Resource.IsLoading

Resource.IsLoading

Brief

Returns true while the resource is loading.

Definition

```
boolean Resource.IsLoading( Resource resource )
```

Arguments

resource - The resource to query.

Return Values

true if the resource is still loading or being requested and false otherwise.

Description

Returns true while the resource is loading.

Examples

```
local isLoading = Resource.IsLoading( res )
```

See Also

- Resource.IsLoaded

Resource.Release

Brief

Releases the specified resource.

Definition

```
Resource.Release( Resource resource )
```

Arguments

resource - The resource to be released.

Return Values

None.

Description

Releases the specified resource.

Examples

```
res = Resource.Find( 'laserGun' )
Resource.Release( res )
```

See Also

- Resource.Find

Resource.Request

Brief

Requests a new resource.

Definition

```
Resource Resource.Request( string|Screenshot content, string type, HttpPostData postData = nil, string
memory = nil, boolean useHttpCompression = false, boolean readBodyOnErrorStatus = false )
```

Arguments

content - The URL of the resource to request or a Screenshot object.type - The type of resource to request. Can be one of the following:
"file"
"texture"
"soundstream"postData - Optional. Post data to send up to the server when making a request.memory - Optional. Valid values are 'host' or 'main'.useHttpCompression - Boolean, indicates if HTTP compression should be used.readBodyOnErrorStatus - Boolean, indicates if the response body should be read if the status code indicates a problem (e.g. for a '410 gone', the body may be used to communicate a reason).

Return Values

A handle to the requested resource, or nil if the resource request failed (v01.45+).

Description

Requests a new resource. Only HTTP/HTTPS downloads are currently supported. Note that images loaded as textures do not currently have control over the compression or resizing of the image for use by PS Home, and as a result may display at a lower quality than the source image. If you wish to control the conversion options used when creating the texture, see Resource.RequestTexture.

Examples

```
-- Example 1: HTTP download of XML
url = "http://www.example.com/data.xml"
xmlData1 = Resource.Request( url, "file" )

-- Example 2: HTTPS download of XML generated by a server-sidew script
url = "https://www.example.com/script.php"
xmlData2 = Resource.Request( url, "file", postData, 'host' )

-- Example 3: After taking a screenshot load it up as a texture
if screenshot:Succeeded() == true then
  screenShotTexture = Resource.Request( screenshot, "texture" )
end

-- Example 4: HTTP download of a texture
url = "http://www.example.com/texture.dds"
xmlData1 = Resource.Request( url, "texture" )
```

See Also

- Screenshot.Create
- Resource.RequestTexture
- Resource.RequestSecure

Resource.RequestSecure

Brief

Requests a secure resource.

Definition

```
Resource Resource.RequestSecure( string contentId, string type, boolean global = false, MemoryType
memType = MemoryType.Main, boolean useHttpCompression = false, boolean readBodyOnErrorStatus = false )
```

Arguments

contentId - The ID (filename) of the secure resource to request. Note that as this call will
ultimately result in an HTTPS request, the contentId is case-sensitive.type - The type of resource to request. Can be one of the following:
"file"global - Request the non-region-specific resource.memType - Memory type to allocate for the resource.useHttpCompression - Boolean, indicates if HTTP compression should be used.readBodyOnErrorStatus - Boolean, indicates if the response body should be read if the status code indicates a problem (e.g. for a '410 gone', the body may be used to communicate a reason).

Return Values

A handle to the requested resource, or nil if the resource request failed.

Description

Requests a new secure resource. There are two secure storage locations, one for the user's current region and one for all regions. Specifying the global flag will request the non-region-specific file rather than the current region's file. If the secure service is not available this function will return nil, scripts should handle this case gracefully.

Examples

```
-- retrieve the global (non-region-specific) version of myfile.xml from secure storage
xmlData = Resource.RequestSecure( "myfile.xml", "file", true )
```

See Also

- Resource.Request

Resource.RequestTexture

Brief

Request a new resource.

Definition

```
Resource Resource.RequestTexture( string|Screenshot content, TextureOptions options =
TextureOptions.Default )
```

Arguments

content - The URL of the resource to request or a Screenshot object.options - The texture load options to use. Can be combined with the + operator. The options available are
TextureOptions.Default - the standard options, as per Resource.Request.
TextureOptions.NoCompression - do not DXT compress the texture
TextureOptions.NoMipmaps - do not generate mipmaps for the texture
TextureOptions.NoSizeLimit - do not cap the major dimension of the texture to 512 texels
TextureOptions.NoPow2 - do not convert the major dimension of the texture to a power of 2 in size
TextureOptions.Image - all of the above 4 options combined

Return Values

A handle to the requested resource, or nil if the resource request failed.

Description

Requests a new texture resource. Only HTTP/HTTPS downloads are currently supported. The default texture load options can cause an image to lose quality due to compression, size limiting and resizing. A script can use this function to specify the exact texture load options to use if required. In normal cases without strict quality requirements, Resource.Request should be preferred.

Examples

```
fullscreenImage = Resource.RequestTexture("http://www.mywebsite.com/HDBackground.jpg",
TextureOptions.Image)
hiResTexture = Resource.RequestTexture("http://www.mywebsite.com/HiResTexture.png",
TextureOptions.NoSizeLimit)
myScreenshot = Resource.RequestTexture(screenshotObject, TextureOptions.NoSizeLimit +
TextureOptions.NoCompression + TextureOptions.NoPow2 + TextureOptions.NoMipmaps)
```

See Also

- Resource.Request

Resource.Run

Brief

Executes the specified resource as a Lua script.

Definition

```
Resource.Run( string resourceName, ResourcePack pack = nil )
Resource.Run( Resource fileResource )
```

Arguments

resourceName - The name of the resource. The type of the resource must be 'lua'.resourcePack - The ResourcePack to search for the specified resourceName, if desired.fileResource - The resource to run. The type of the resource must be 'lua'.

Return Values

None.

Description

Executes the specified resource as a Lua script.

Examples

```
Resource.Run( "utils.lua" )
```

See Also

- LoadLibrary

Resource.StartLoading

Brief

Starts loading the resource if it is not already loaded.

Definition

```
boolean Resource.StartLoading( Resource resource, number memorySlotId )
```

Arguments

resource - The resource to be loaded.memorySlotId - The id of the memory slot to use or 0 for the default budget

Return Values

true if successful and false otherwise.

Description

Starts loading the resource if it is not already loaded. If a callback is specified, it will be passed a boolean value indicating whether the load succeeded.

Examples

```
res = Resource.Find( 'laserGun' )
Resource.StartLoading( res )
res2 = Resource.Find( 'mapSection2' )
Resource.StartLoading( res2, 2 )        -- load into memory slot 2
```

See Also

None

ResourcePack

ResourcePack.DebugGetLoadError

Brief

Get the error code for a load error.

Definition

```
RpLoadError ResourcePack.DebugGetLoadError( ResourcePack pack )
```

Arguments

pack - Resource Pack object.

Return Values

An RpLoadError value indicating the reason for the load failure:
RpLoadError.None – no error
RpLoadError.MountFail – object package could not be mounted
RpLoadError.OfflineFail – offline object could not be loaded
RpLoadError.DownloadFail – download of object package failed
RpLoadError.ResourcesFail – resources XML did not load or parse correctly
RpLoadError.ResourceAddFail – one of the object resources did not load
RpLoadError.LocalisationFail – localization XML did not load or parse correctly
RpLoadError.CreateDefFail – object definition could not be created
RpLoadError.ObjectDefFail – object XML did not load or parse correctly
RpLoadError.CreateFail – object create failed
RpLoadError.MakeFail – object type could not be found
RpLoadError.Component – one of the components failed to load (should not occur for RPs)

RpLoadError.NotAResourcePack – load references an object that is not an RP
RpLoadError.PermissionDenied – either requesting object or RP doesn't list object in permissions

Description

If GetLoadState indicates an error has occurred, this function can be used to retrieve the error state. This function is not available in live content and is not intended for display to a user, but can assist when developing the script.

Examples

```
myResPack = ResourcePack.Load( "DEADBEEF-00000000-00000000-00000000" )
if (not myResPack) then
    local e = myResPack:GetLoadError()

end
...
while (myResPack:GetLoadState() ~= RpLoadState.Loaded and myResPack:GetLoadState() ~=
RpLoadState.Error) do
    coroutine.yield()
end
if (myResPack:GetLoadState() == RpLoadState.Error) then
    local e = myResPack:GetLoadError()
end
```

See Also

- ResourcePack.GetLoadState
- ResourcePack.Load

ResourcePack.GetLoadProgress

Brief

Get the progress percentage of the current load state.

Definition

```
number ResourcePack.GetLoadProgress( ResourcePack pack )
```

Arguments

pack - Resource Pack object.

Return Values

A number in the range [0, 1] representing the proportion of the current load state that is complete.

Description

The Downloading and Loading states will each take a period of time to complete. A script should indicate to the user the progress of these steps individually so the load process is clearly understood. If this function is called for other states the value is likely to be fixed at 0 or 1 and should not be displayed to the user.

Examples

```
if ( loadState == RpLoadState.Downloading ) then
    myDownloadProgressBarGui:SetProgress( myResPack:GetLoadProgress() * 100.0 )     -- %
elseif ( loadState == RpLoadState.Loading ) then
    myLoadProgressBarGui:SetProgress( myResPack:GetLoadProgress() * 100.0 )     -- %
end
```

See Also

- ResourcePack.GetLoadState
- ResourcePack.Load

ResourcePack.GetLoadState

Brief

Get the current load state of a ResourcePack.

Definition

```
RpLoadState ResourcePack.GetLoadState( ResourcePack pack )
```

Arguments

pack - Resource Pack object.

Return Values

A RpLoadState enum indicating the current load state:
RpLoadState.Created
RpLoadState.Queued
RpLoadState.Downloading
RpLoadState.Loading
RpLoadState.Loaded
RpLoadState.Error

Description

The load of a ResourcePack is an asynchronous process which may take some time to complete. This function will indicate the current state of the ResourcePack load. Once the Loaded status is returned the object is ready to use. If this function indicates Error, the script can use GetLoadError to find out why the load failed. The Downloading and Loading states can be further queried for their progress using the GetLoadProgress function.

Examples

```
while (myResPack:GetLoadState() ~= RpLoadState.Loaded and myResPack:GetLoadState() ~=
RpLoadState.Error) do
    coroutine.yield()
end
if (myResPack:GetLoadState() == RpLoadState.Error) then
    print("ResourcePack failed to load!")
end
```

See Also

- ResourcePack.Load
- ResourcePack.GetLoadProgress

ResourcePack.IsLoading

Brief

Returns true if resource pack is currently downloading or loading.

Definition

```
ResourcePack.IsLoading(ResourcePack pack)
```

Arguments

pack - The ResourcePack object.

Return Values

True if the resource pack is download or loading, false otherwise.

Description

This function will check the load state and indicate whether the resource pack is downloading/loading. Resource packs that have completed loading or failed to load will return false.

Examples

```
if (myResPack:IsLoading()) then
    RenderProgressBar(myResPack:GetLoadState(), myResPack:GetLoadProgress())        -- user function
end
```

See Also

- ResourcePack.GetLoadState

ResourcePack.Load

Brief

Creates a ResourcePack and starts downloading the object.

Definition

```
ResourcePack ResourcePack.Load( string objectId, number memorySlotId = 0 )
```

Arguments

objectId - ID of resource pack object to load.memorySlotId - ID number of memory slot to load the resource pack into.

Return Values

A new ResourcePack object, or nil if object not found or could not be loaded.

Description

Initiates a background load of the specified ResourcePack object. The object ID must relate to a ResourcePack object for which this object has load permission, other object IDs will be refused. The load is asynchronous, so the script should query the progress using GetLoadState and GetLoadProgress, and display appropriate feedback to the user while the load process is taking place. If nil is returned the object ID is likely to be invalid, and the load request should not be retried. The script should call Unload when the ResourcePack is no longer required, ensuring that no resources remain in use.

Examples

```
myResPack = ResourcePack.Load( "11111111-11111111-11111111-11111111" )
if (not myResPack) then
    print("Error loading resource pack!")
    return
end
```

See Also

- ResourcePack.GetLoadState
- ResourcePack.GetLoadProgress
- ResourcePack.Unload

ResourcePack.Unload

Brief

Unloads the ResourcePack and all associated resources.

Definition

```
ResourcePack.Unload( ResourcePack pack )
```

Arguments

pack - Resource Pack object.

Return Values

None.

Description

Unloads the ResourcePack object and all resources referenced by it. Note that the script should ensure that no resources owned by the ResourcePack remain in use by PS Home as the client will not check for this. The ResourcePack Lua object remains valid and can be used to load a new ResourcePack. Garbage collected ResourcePack Lua objects will call this function automatically.

Examples

```
if (loadingNextLevel) then
    myResPack:Unload()
end
```

See Also

- ResourcePack.Load

Rewards

Rewards.AddTicket

Brief

Adds a reward ticket corresponding to the specified object ID to the list of tickets awaiting collection by the user.

Definition

```
boolean Rewards.AddTicket( string objectId, string callBackString = nil )
```

Arguments

objectId string - The ObjectID for which the ticket is allocated.callBackString - The optional call back string to run once the ticket has been processed.

Return Values

False if immediately failed to add the ticket, such as it being in the inventory already and true if successfully added. It may fail later though.

Description

This function assigns a reward ticket to a player which can award an object to a player instantly or can be used to collect a reward at a later point. The type of ticket given is determined by the entitlement_id fields in the object's metadata (edited using the HDK's Object Editor application). These are set per region and can be either AUTOMATIC_REWARD (instantly awarded), or LUA_REWARD (awarded using Rewards.CollectTicket()). If a callback is specified you can use Rewards.GetCurrentFocusObjectId and Rewards.GetCurrentFocusState inside the callback to check the result.

The PS Home client must be run in online mode in order for this function to work.

Examples

```
Rewards.AddTicket( "12345678-12345678-12345678-12345678" )
```

See Also

- Rewards.DoesHaveTicket
- Rewards.CollectTicket

Rewards.CollectTicket

Brief

Attempts to collect a reward ticket that must have been previously added to the user's reward ticket list.

Definition

```
boolean Rewards.CollectTicket( string objectId, string param, string callBackString = nil )
```

Arguments

objectId - The object ID of the object to collect as a reward.param - Reserved for future use.callBackString - Optional lua string thats run once the ticket has been collected or an error occurs in collection.

Return Values

true if call succeeds, or false if there is no ticket to collect.

Description

This function asks the PS Home Reward System to attempt to collect the reward ticket corresponding to the specified object ID. This will add the object to the user's permanent inventory. Note that PS Home must first retrieve data regarding the object from its content servers to verify that the item can be added, and so it is likely that the object will not appear in the user's inventory immediately. There are many conditions (such as the object not having been set to be rewarded through Lua script) that can mean the object is not actually added to the inventory. A return value of true from this function only indicates that the initial call was successful, it does not indicate that the user now owns the item. This can be verified by querying LocalPlayer.IsInInventory on successive frames. The PS Home client must be run in online mode in order for this function to work.

Examples

```
Rewards.CollectTicket( "12345678-12345678-12345678-12345678" )
```

See Also

- Rewards.DoesHaveTicket
- Rewards.AddTicket

Rewards.DoesHaveTicket

Brief

Checks whether a Reward Ticket for the specified ObjectID is currently awaiting collection.

Definition

```
boolean Rewards.DoesHaveTicket( string objectId )
```

Arguments

objectId - The object ID to check for.

Return Values

true if the reward ticket for the specified object ID is currently in the list awaiting collection.

Description

This function is used to determine when the user has been awarded a reward ticket but has not yet collected it. For example, an external title may award the ticket using a REWARDS.XML written to the PLAYSTATION3's hard drive, based on a certain condition within the external title. The title's developer could then develop a LUA script for PS Home, that checks if the reward ticket has been awarded, then shows appropriate feedback to the user before collecting the reward ticket. The PS Home client must be run in online mode in order for this function to work.

Examples

```
rewardObjId = "12345678-12345678-12345678-12345678"
if ( Rewards.DoesHaveTicket( rewardObjId ) ) then
    print("I have that ticket!")
end
```

See Also

- Rewards.AddTicket
- Rewards.CollectTicket

Rewards.GetCurrentFocusObjectId

Brief

Returns the object ID for the callback being called.

Definition

```
string Rewards.GetCurrentFocusObjectId()
```

Arguments

None

Return Values

The object ID in focus.

Description

Returns the object ID for the callback being called. Only valid when inside a callback. The PS Home client must be run in online mode in order for this function to work.

Examples

```
function RewardCallBack()
    local id = Rewards.GetCurrentFocusObjectId()
    if id == g_laserGun then
        local state = Rewards.GetCurrentFocusState()
        if state == 'collected' or state == 'added' then
            ShowLaserMessageBox()
        else
            ShowNoLaserForYouError()
        end
    end
end
```

See Also

- Rewards.CollectTicket
- Rewards.GetCurrentFocusState

Rewards.GetCurrentFocusState

Brief

Returns the state for the callback being called.

Definition

```
string Rewards.GetCurrentFocusState()
```

Arguments

None

Return Values

The state of callback in focus.

Description

Returns the state for the callback being called. Only valid when inside a callback. States returned are:
"collected" - The ticket has been collected.
"added" - The ticket has been added and ready for collection.
"noinfo" - There is no reward information for the object.
"failed" - A general error occurred.
"failed_automatic" - The collection failed as the reward is automaticly collected.
The PS Home client must be run in online mode in order for this function to work.

Examples

```
function RewardCallBack()
    local id = Rewards.GetCurrentFocusObjectId()
    if id == g_laserGun then
        local state = Rewards.GetCurrentFocusState()
        if state == 'collected' or state == 'added' then
            ShowLaserMessageBox()
        else
            ShowNoLaserForYouError()
        end
    end
end
```

See Also

- Rewards.GetCurrentFocusObjectId

RtGame

RtGame.AddCustomExitMenuOption

Brief

Add a custom menu item to the Exit Menu for the mini-game.

Definition

```
RtGame.AddCustomExitMenuOption(string optionText)
```

Arguments

optionText - The localized string to display. Note: This must contain the "[SQUARE]" tag to denote the SQUARE button selects this option. This will almost certainly be at the start of the text, and likely include a space after it, to separate it from the rest of the text. E.g. "[SQUARE] Settings".

Return Values

None.

Description

This enables the custom menu option in the mini-game's Exit Menu, and sets the text for that option. The text must be localized before input. Only one custom option can be added. Successive calls to this function can be used to alter the text displayed.

Examples

```
local customText = Object.GetMe():GetLocalizedText("myCustomOptionText")
RtGame.AddCustomExitMenuOption(myCustomOptionText)
```

See Also

- RtGame.RemoveCustomExitMenuOption
- RtGame.IsAbleToChangeCustomExitMenuOption

RtGame.CreateGame

Brief

Creates an RtGame session manually.

Definition

```
RtGame.CreateGame()
```

Arguments

None

Return Values

None.

Description

Creates an RtGame session. When the session is set up, a unique "room ID" is generated. This is available via the function RtGame.GetRoomId.

Examples

```
roomId = BigInt.Create(64)
if (not RtGame.IsJoined()) then
    RtGame.CreateGame()
    creating = true
end

function OnUpdate()
    if (creating and RtGame.IsJoined()) then
        RtGame.GetRoomId(roomId)
        creating = false
    end
end
```

See Also

- RtGame.JoinGame
- RtGame.IsJoined
- RtGame.GetRoomId

RtGame.CreateGameObject

Brief

Creates a RtGameObject instance.

Definition

```
RtGameObject RtGame.CreateGameObject( string objectType )
```

Arguments

objectType - The name of the object type defined within the network_definitions_xml file

Return Values

An RtGameObject instance, or nil if failed.

Description

Creates a RtGameObject instance. Once an object is created, initial values should be set before registering it for network replication. The RtGameObject fields can be accessed in normal Lua table format.

Examples

```
if IsSessionMaster() then
    session = RtGame.CreateGameObject( "gamestate" )
    session.gameMode = 0
    session.timeLeft = 10.0
    session.teamAScore = 0
    session.teamBScore = 0
    RtGame.ReplicateToNet( session )
end
```

See Also

- RtGame.ReplicateToNet

RtGame.DestroyGameObject

Brief

Destroys a RtGameObject instance.

Definition

```
RtGame.DestroyGameObject( RtGameObject rtGameObject )
```

Arguments

rtGameObject - the RtGameObject instance to be destroyed.

Return Values

none.

Description

Destroys a RtGameObject instance. Only RtGame objects owned by the local player can be destroyed.

Examples

```
RtGame.DestroyGameObject(myGameObject)
```

See Also

- RtGame.CreateGameObject

RtGame.GetConnectionInfo

Brief

Retrieves connection quality information for a member.

Definition

```
number RtGame.GetConnectionInfo( number memberId, number connectionInfoType )
```

Arguments

memberId - A player's member ID.connectionInfoType - Connection information value to obtain (see details).

Return Values

Connection info value requested, or nil on failure.

Description

Retrieves connection quality information for a member. connectionInfoType should be specified using the following enums:

- RtConnInfo.Rtt - Round-trip time (microseconds)
- RtConnInfo.Bandwidth - Estimate bandwidth (bits/second)
  Only one piece of information may be queried for each call to GetConnectionInfo.

Examples

```
for i = 1, RtGame.GetPlayerCount() do
    local memberId = RtGame.GetPlayer(i)
    if (memberId ~= nil) then
        local user = RtGame.GetUserFromMemberId(memberId)
        local rtt  = RtGame.GetConnectionInfo( memberId, RtConnInfo.Rtt )
        if (rtt ~= nil and user ~= nil) then
            print( string.format("Player %d: Name=%d, rtt=%d", i, user:GetName(), rtt) )
        end
    end
end
```

See Also

None

RtGame.GetMaxPlayers

Brief

Retrieves the maximum number of players for this session.

Definition

```
number RtGame.GetMaxPlayers()
```

Arguments

None

Return Values

Maximum number of players.

Description

Retrieves the maximum number of players for this session.

Examples

```
print( string.format("%d/%d players joined", RtGame.GetPlayerCount(), RtGame.GetMaxPlayers()) )
```

See Also

None

RtGame.GetMemberIdFromUser

Brief

Retrieves a member ID given a User object.

Definition

```
number RtGame.GetMemberIdFromUser( User user )
```

Arguments

user - A user to look up.

Return Values

MemberId associated with user, or nil if the user could not be located in the RT Game session.

Description

Retrieves a member ID given a User object if the user is part of the RT Game session.

Examples

```
friends = LocalPlayer.GetFriends()
for i, friend in ipairs( friends ) do
    local memberId = RtGame.GetMemberIdFromUser(friend)
    if (memberId ~= nil) then
        print( string.format("Friend %s is in game, with member ID %d", friend:GetName(), memberId) )
    end
end
```

See Also

None

RtGame.GetNumTokens

Brief

Retrieves the number of tokens available for this game.

Definition

```
number RtGame.GetNumTokens()
```

Arguments

None

Return Values

The maximum number of tokens.

Description

Retrieves the number of tokens available for this game. This will be the value set in the realtime_game component within the Object Editor.

Examples

```
print("Num tokens =", RtGame.GetNumTokens())
```

See Also

None

RtGame.GetOwnerMemberId

Brief

Retrieves the session owner's member ID.

Definition

```
number RtGame.GetOwnerMemberId()
```

Arguments

None

Return Values

Session owner's member ID, or nil if not in session or there is no owner.

Description

Retrieves the session owner's member ID.

Examples

```
function IsSessionOwner()
    return localPlayerMemberId == RtGame.GetOwnerMemberId()
end
```

See Also

None

RtGame.GetPlayer

Brief

Retrieves a member ID for a player in the session.

Definition

```
number RtGame.GetPlayer( number playerNumber )
```

Arguments

playerNumber - The index of the player slot to query, in the range [1, RtGame.GetPlayerCount()]

Return Values

memberId for player at given slot

Description

Retrieves a member ID for a player in the session.

Examples

```
for i = 1, RtGame.GetPlayerCount() do
    local memberId = RtGame.GetPlayer(i)
    print( string.format("Player %d: ID=%d", i, memberId) )
end
```

See Also

None

RtGame.GetPlayerCount

Brief

Retrieves the number of players currently participating in this session.

Definition

```
number RtGame.GetPlayerCount()
```

Arguments

None

Return Values

Number of players.

Description

Retrieves the number of players currently participating in this session.

Examples

```
print( string.format("%d/%d players joined", RtGame.GetPlayerCount(), RtGame.GetMaxPlayers()) )
```

See Also

None

RtGame.GetRoomId

Brief

Get the unique room ID of the currently joined RtGame session.

Definition

```
RtGame.GetRoomId(BigInt<64> roomId)
BigInt<64> RtGame.GetRoomId()
```

Arguments

roomId - a 64-bit BigInt in which to store the room ID.

Return Values

A 64-bit BigInt of the room ID if no BigInt was passed in.

Description

If the user is currently joined to an RtGame session, this function will return the room ID of the session.

Examples

```
roomId = BigInt.Create(64)
if (RtGame.IsJoined()) then
    RtGame.GetRoomId(roomId)
end
```

See Also

- RtGame.IsJoined
- RtGame.JoinGame

RtGame.GetSentInstanceTime

Brief

Retrieves the instance time of when the data was sent.

Definition

```
number, number RtGame.GetSentInstanceTime()
```

Arguments

None

Return Values

Instance time, split into 2 number values.

Description

Retrieves the instance time of when the data was sent for data receipt callbacks. See GetInstanceTime() for more details. This function is only valid within the following functions: on_object_replicated on_object_updated on_message

Examples

```
b,t = RtGame.GetSentInstanceTime()
print( string.format( "The instance time in hex is %#.2x%.6x", t, b ) )
```

See Also

- GetInstanceTime

RtGame.GetTokenOwnerMemberId

Brief

Queries the ownership of a token.

Definition

```
number RtGame.GetTokenOwnerMemberId( number tokenId )
```

Arguments

tokenId - Token number to query.

Return Values

Member ID of the token's owner, nil if the token is unowned.

Description

Queries the ownership of a token. Please note that the return value only reflects the current token state on the local client, an unowned token does not guarantee that a request of that token will succeed.

Examples

```
if (localMemberId == RtGame.GetTokenOwnerMemberId( weaponTokenId )) then
    print("You have the weapon!")
end
```

See Also

- RtGame.RequestToken
- RtGame.ReleaseToken

RtGame.GetUserFromMemberId

Brief

Retrieves a User object associated with a member ID.

Definition

```
User RtGame.GetUserFromMemberId( number memberId )
```

Arguments

memberId - A player's member ID.

Return Values

User object associated with a memberId, or nil on failure.

Description

Retrieves a User object associated with a memberId.

Examples

```
for i = 1, RtGame.GetPlayerCount() do
    local memberId = RtGame.GetPlayer(i)
    if (memberId ~= nil) then
        local user = RtGame.GetUserFromMemberId(memberId)
        if (user ~= nil) then
            print( string.format("Player %d: Name=%d", i, user:GetName()) )
        end
    end
end
```

See Also

None

RtGame.IsAbleToChangeCustomExitMenuOption

Brief

Determines if a custom menu option can currently be added to the Exit Menu.

Definition

```
RtGame.IsAbleToChangeCustomExitMenuOption()
```

Arguments

None

Return Values

Whether a custom menu option can currently be added to the Exit Menu.

Description

Changing the custom menu option while the Exit Menu is visible is not user fiendly and is therefore restricted. This function can be used to determine when 'AddCustomExitMenuOption' and 'RemoveCustomExitMenuOption' will fail.

Examples

```
RtGame.IsAbleToChangeCustomExitMenuOption()
```

See Also

- RtGame.AddCustomExitMenuOption
- RtGame.RemoveCustomExitMenuOption

RtGame.IsJoined

Brief

Indicates whether the local user is joined to an RtGame session.

Definition

```
boolean RtGame.IsJoined()
```

Arguments

None

Return Values

True if the local user is joined to an RtGame session, or false otherwise.

Description

Indicates whether the local user is joined to an RtGame session.

Examples

```
roomId = BigInt.Create(64)
if (not RtGame.IsJoined()) then
    RtGame.CreateGame()
    creating = true
end

function OnUpdate()
    if (creating and RtGame.IsJoined()) then
        RtGame.GetRoomId(roomId)
        creating = false
    end
end
```

See Also

- RtGame.CreateGame
- RtGame.JoinGame
- RtGame.GetRoomId

RtGame.IsSessionOpen

Brief

Queries whether the session is open.

Definition

```
boolean RtGame.IsSessionOpen()
```

Arguments

None.

Return Values

true if session is open, false otherwise.

Description

Queries whether the session is open. This function is valid only for the session owner. After a session ownership change event, the new session owner may use this function to query the session state.

Examples

```
function OnSessionOwnerChange( newOwnerId )
   print( "The session is open = ", tostring( RtGame.IsSessionOpen() ) )
end
```

See Also

- RtGame.SetSessionClosed

RtGame.JoinGame

Brief

Joins an existing RtGame session.

Definition

```
RtGame.JoinGame(BigInt<64> roomId)
```

Arguments

roomId - a 64-bit BigInt containing the room to join.

Return Values

None.

Description

Joins the local user to an existing RtGame session with the specified room ID. The user should not be connected to an RtGame session already.

Examples

```
-- roomId has been obtained from session owner...
if (not RtGame.IsJoined()) then
    RtGame.JoinGame(roomId)
    joining = true
end

function OnUpdate()
    if (joining and RtGame.IsJoined()) then
        joining = false
    end
end
```

See Also

- RtGame.CreateGame
- RtGame.GetRoomId

RtGame.Leave

Brief

Leave the session that the local user is part of.

Definition

```
RtGame.Leave()
```

Arguments

None

Return Values

None

Description

Leave the session that the local user is part of.

Examples

```
if (userWantsToQuit) then
    RtGame.Leave()
end
```

See Also

None

RtGame.ReleaseToken

Brief

Requests that ownership of a token is released.

Definition

```
boolean RtGame.ReleaseToken( number tokenId )
```

Arguments

tokenId - Token number to release.

Return Values

True if request submitted successfully, false if local user is not the owner of the token.

Description

Requests that ownership of a token is released. Token release requests are sent to the session owner. Use the on_token_ownership_change callback to detect if the request was successful. Only the token's owner or the session owner can request a token's release.

Examples

```
if (dropWeapon) then
    RtGame.ReleaseToken( weaponTokenId )
end
```

See Also

- RtGame.RequestToken

RtGame.RemoveCustomExitMenuOption

Brief

Remove the custom menu item from the Exit Menu for the mini-game (if one has been added).

Definition

```
RtGame.RemoveCustomTargetMenuOption(string optionText)
```

Arguments

None

Return Values

None.

Description

This removes the custom menu option in the mini-game's Exit Menu, if one has previously been added.

Examples

```
RtGame.RemoveCustomExitMenuOption()
```

See Also

- RtGame.AddCustomExitMenuOption
- RtGame.IsAbleToChangeCustomExitMenuOption

RtGame.ReplicateToNet

Brief

Marks an object for auto replication.

Definition

```
RtGame.ReplicateToNet( RtGameObject rtGameObject )
```

Arguments

rtGameObject - the RtGameObject instance to be replicated.

Return Values

None.

Description

Marks an object for auto replication. Once an object is created, it remains locally until this function is called. Once replicated to the network, the object will continue to be updated until it is destroyed or the player leaves the session.

Examples

```
if IsSessionMaster() then
    session = RtGame.CreateGameObject( "gamestate" )
    session.gameMode = 0
    session.timeLeft = 10.0
    session.teamAScore = 0
    session.teamBScore = 0
    RtGame.ReplicateToNet( session )
end
```

See Also

RtGame.RequestToken

Brief

Requests ownership of a token.

Definition

```
RtGame.RequestToken( number tokenId )
```

Arguments

tokenId - Token number to request.

Return Values

None.

Description

Requests ownership of a token. Tokens are requested from the session owner. Use the on_token_ownership_change callback to detect if the request was successful. As the session owner will have the advantage with token requests, it may be worth adding mechanisms within your script to make your game more fair. For example, delaying requests for the session owner.

Examples

```
RtGame.RequestToken( weaponTokenId )
```

See Also

- RtGame.ReleaseToken

RtGame.ReturnJoinGameResult

Brief

Returns the result for the custom join game step of the joining process.

Definition

```
RtGame.ReturnJoinGameResult( boolean result )
```

Arguments

result - true to allow the user to join, or false otherwise.

Return Values

None.

Description

When a callback function in on_join_game_request is specified, the system will wait for this call before proceeding with the game join process. Note that this call need not be made immediately within the callback, the script may perform asynchronous processing if necessary and call this function at a later point. In this case the script should provide an indication to the user that the join process is still in progress. See the on_local_player_join_request field within the realtime_game component of the object.

Examples

```
function OnLocalPlayerJoinRequest()
    RtGame.ReturnJoinGameResult(IsGameJoinable())
end
```

See Also

None

RtGame.SendMessage

Brief

Marks a message for sending

Definition

```
RtGame.SendMessage( string messageType, table message, number memberId = -1 )
```

Arguments

messageType - The name of the message as specified in the network_definition_file xml.message - The lua table containing all the message fields.memberId - MemberId of user to receive the message. This is an optional argument, omitting it or specifying -1 will send the message to all members.

Return Values

None.

Description

Marks a message for sending. Note: The message table does not need to persist after the call to this function.

Examples

```
local msg = { id = missleType, damage = 23, vX = getFacingX(), vY = getFacingY(), vZ = getFacingZ() }
RtGame.SendMessage( "fireMissile", msg )
```

See Also

None

RtGame.SetSessionClosed

Brief

Sets the session open/close state.

Definition

```
RtGame.SetSessionClosed( boolean close )
```

Arguments

close - True to close session, false to open.

Return Values

None.

Description

Sets the session open/close state. Once the session is closed, no other users may join. Note that this is a server request and therefore does not take immmediate effect, use the on_session_access_change callback to determine when the action is complete. This function is valid only for the session owner.

Examples

```
if (raceIsReadyToStart) then
    RtGame.SetSessionClosed(true)
end
```

See Also

None

RtGameObject

RtGameObject.GetInstanceId

Brief

Queries the object's owner.

Definition

```
number RtGameObject.GetInstanceId( RtGameObject obj )
```

Arguments

obj - The RT Game Object to query.

Return Values

Object's instance ID.

Description

Queries the object's instance ID. The ID returned is globally unique to the creator, across all objects of any type. This ID in combination with the owner will uniquely identify a given RT Game Object.

Examples

```
function OnObjectReplicated( rtGameObject )

    local owner = rtGameObject:GetOwner()
    local instanceId = rtGameObject:GetInstanceId()
    print( string.format("OnObjectReplicated = %s, owner = %s, instnaceId = %d",
rtGameObject:GetName(), owner:GetName(), instanceId) )

end
```

See Also

None

RtGameObject.GetName

Brief

Queries the object's name.

Definition

```
string RtGameObject.GetName( RtGameObject obj )
```

Arguments

obj - The RT Game Object to query.

Return Values

Object's name.

Description

Queries the object's name. The name is the string specified in the 'name' attribute when defining this object in the network_definition_file file given in the Object Editor.

Examples

```
function OnObjectReplicated( rtGameObject )
    print( string.format("OnObjectReplicated = %s", rtGameObject:GetName()) )
end
```

See Also

None

RtGameObject.GetOwnerMemberId

Brief

Queries the object's owner.

Definition

```
number RtGameObject.GetOwnerMemberId( RtGameObject obj )
```

Arguments

obj - The RT Game Object to query.

Return Values

Retrieves the memberId of the object's owner.

Description

Queries the object's owner.

Examples

```
function OnObjectReplicated( rtGameObject )

    ownerId = rtGameObject:GetOwnerMemberId()
    print( string.format("OnObjectReplicated = %s, ownerId = %d", rtGameObject:GetName(), ownerId) )

end
```

See Also

None

RtGameObject.GetOwnershipType

Brief

Queries the object's ownership type.

Definition

```
number RtGameObject.GetOwnershipType( RtGameObject obj )
```

Arguments

obj - The RT Game Object to query.

Return Values

Object's ownership type (see enums below)

Description

Queries the object's ownership type, as defined by the 'type' attribute of the object's defninition in the network defnition xml.

- RtObjOwnership.Personal
- RtObjOwnership.Session

Examples

```
function OnObjectReplicated( rtGameObject )

    local owner = rtGameObject:GetOwner()

    if rtGameObject:GetOwnershipType() == RtObjOwnership.Personal then

        print( string.format("OnObjectReplicated = %s, owner = %s, Personal obj",
rtGameObject:GetName(), owner:GetName()) )

    else

        print( string.format("OnObjectReplicated = %s, owner = %s, Session obj",
rtGameObject:GetName(), owner:GetName()) )

    end

end
```

See Also

None

SaveData

SaveData.Create

Brief

Creates a new save data request object.

Definition

```
SaveData.Create()
```

Arguments

None.

Return Values

A new instance of a save data request object.

Description

Creates a new save data request object.

Examples

```
saveDataRequest = SaveData.Create()
```

See Also

SaveData.GetState

Brief

Returns the current status of the request.

Definition

```
SaveData.GetState( SaveData request )
```

Arguments

request - The request to check the state for

Return Values

One of the following states:
0 - Idle. The request has not started or has completed without any problems
1 - The request is in progress
2 - The request is in progress (but is in a new internal state, scripts should treat this state no differently from 1)
-1 - The request has failed
-2 - The was no data to load (not technically an error as this will happen when the script is run for the first time)

Description

Returns the current status of the request. Scripts can monitor the status to see when a request completes and if it was successful or not.

Examples

```
local state = SaveData.GetState( saveDataRequest )
```

See Also

SaveData.StartLoading

Brief

Submits a save data load request.

Definition

```
number SaveData.StartLoading( SaveData request, string dataId, number type, MemoryContainer container
)
```

Arguments

request - The request to submitsaveId - The data identifier. Must match the id used for saving.type - Reserved, must be 0container - The container to load the data into

Return Values

0 on success or one of the following errors:
-1 - Invalid params.
-2 - This request object is already busy.
-3 - This is a duplicate of another request in progress (only affects save request).
-4 - The container is too large to save. Save sizes are limited to 4kb.
-5 - Loading and saving has been disable due to an earlier problem.

Description

Submits a save data load request. Multiple load requests for the same data can be submitted providing that there are no in progress save requests. Please contact your Playstation(R)Home Regional Account Manager to request a save data key. Note: save data keys are currently limited and use of the save data API will initially be on a restricted basis.

Examples

```
local result = SaveData.StartLoading( saveDataRequest, "XXXXXX", 0, container )
```

See Also

SaveData.StartSaving

Brief

Submits a save data save request.

Definition

```
number SaveData.StartSaving( SaveData request, string dataId, number type, MemoryContainer container,
number size = 0 )
```

Arguments

request - The request to submitsaveId - The data identifier. Must match the id used for loadingtype - Reserved, must be 0container - The container to save the data fromsize - The number of bytes to be saved. If not specified, the UsedSize of the memory container will be used if set. Otherwise, the total memory container size will be used.

Return Values

0 on success or one of the following errors:
-1 - Invalid params.
-2 - This request object is already busy.
-3 - This is a duplicate of another request in progress (only affects save request).
-4 - The container is too large to save. Save sizes are limited to 4kb.
-5 - Loading and saving has been disable due to an earlier problem.

Description

Submits a save data save request. Only a single save request for a single dataId can be in progress at a time. If there are any other loads or saves in progress, the script will need to wait for those to complete first. Please contact your Playstation(R)Home Regional Account Manager to request a save data key. Note: save data keys are currently limited and use of the save data API will initially be on a restricted basis.

Examples

```
local result = SaveData.StartSaving( saveDataRequest, "XXXXXX", 0, container )
```

See Also

Scene

Scene.FindEntity

Brief

Finds an entity that was created from the scene

Definition

```
Entity Scene.FindEntity( string name )
```

Arguments

name - The name of the entity as set in the HDK's Scene Editor application.

Return Values

The entity if it is found and nil otherwise.

Description

Finds an entity that was created from the scene, (i.e. one added to the world as part of the .scene file)

Examples

```
anotherDancingMonkey = Scene.FindEntity( "DancingMonkey342" )
Entity.SetAnim( anotherDancingMonkey, "monkey_excited" )
```

See Also

Scene.FindScreen

Brief

Finds a screen in the active scene.

Definition

```
Screen Scene.FindScreen( string name )
```

Arguments

name - The name of the screen as set in the HDK's Scene Editor application.

Return Values

The screen matching the specified name or nil if the screen is not found.

Description

Finds a screen in the active scene.

Examples

```
screen = Scene.FindScreen( "MyVideoScreen" )
if screen ~= nil then
    Screen.SetContent( screen, "http://www.example.com/myvideo.mp4" )
end
```

See Also

- Screen.SetContent

Scene.FindTriggerVolume

Brief

Finds a trigger volume that was created from the scene

Definition

```
TriggerVolume Scene.FindTriggerVolume( string name )
```

Arguments

name - The name of the trigger volume as set in the HDK's Scene Editor application.

Return Values

The trigger volume if it is found and nil otherwise.

Description

Finds a trigger volume that was created from the scene, (i.e. one added to the world as part of the .scene file)

Examples

```
trigVol = Scene.FindTriggerVolume( "sceneTriggerVolume1" )
TriggerVolume.AddCallback( trigVol, MySceneTriggerVolumeCallback )
```

See Also

- TriggerVolume.AddCallback

Scene.GetGraphicsEngineParam

Brief

Get the value of a parameter from the graphics engine.

Definition

```
Scene.GetGraphicsEngineParam(paramName)
```

Arguments

paramName - a valid graphics engine param name. Valid values are:
GraphicsEngine.ShadowEnable
GraphicsEngine.ShadowQuality
GraphicsEngine.SunlightColor
GraphicsEngine.SunlightDir
GraphicsEngine.ColorBalance
GraphicsEngine.Saturation
GraphicsEngine.Contrast
GraphicsEngine.ShadowPlaneDistance
GraphicsEngine.ShadowPlaneMultiplier
GraphicsEngine.ShadowPlaneCount
GraphicsEngine.ShadowDarkness
GraphicsEngine.AlphaSortEnable
GraphicsEngine.LightProbeTint
GraphicsEngine.GlowType
GraphicsEngine.GlowColor
GraphicsEngine.GlowBlendMode

Return Values

The current value of the graphics engine parameter in question.

Description

For details of the types returned by this function, see Scene.SetGraphicsEngineParam.

Examples

```
print(Scene.GetGraphicsEngineParam(GraphicsEngine.SunlightColor)) -- output is "[1.64, 1.52, 1.08, 0]"
print(Scene.GetGraphicsEngineParam(GraphicsEngine.ShadowEnable)) -- output is "true"
```

See Also

- Scene.SetGraphicsEngineParam

Scene.GetInstanceId

Brief

Returns a unique id for the scene instance the user is currently in

Definition

```
string Scene.GetInstanceId()
```

Arguments

None.

Return Values

string - The unique scene instance id

Description

Returns a unique id for the scene instance the user is currently in

Examples

```
sceneInstaceId = Scene.GetInstanceId()
```

See Also

None

Scene.GetLaunchParam

Brief

Returns one of the text parameters that may have been passed to PS Home when launched.

Definition

```
string Scene.GetLaunchParam( number index = 1 )
```

Arguments

index - Index of the parameter to fetch (1 - 16).

Return Values

The indexed string that PS Home was launched with, or "none" if none.

Description

This function allows your script to query optional values passed to Home through a configuration file either set up by SCE regional teams or generated from a game launched by Home. This allows you to gamelaunch from Home into your external title, then when the user selects to return to PS Home they are directed to different spaces depending on whether they won or lost (for instance). Note that these values are only valid in the first scene the user enters after returning to Home. See external documentation for more details on setting up this system.

Examples

```
param = Scene.GetLaunchParam( 1 )
if ( param ~= "none" ) then
    if ( param == "winner" ) then       -- custom string passed to PS Home by title
        print("You won!")
    else
        print("You lost.")
    end
end
```

See Also

None

Scene.GetLayerCollisionDefault

Brief

Query the default enabledness of collision detection between collision of the specified layers.

Definition

```
boolean Scene.GetLayerCollisionDefault( enum layerA, enum layerB )
```

Arguments

layerA - The first layer.layerB - The second layer.

Return Values

true if collision detection is enabled by default, false otherwise.

Description

Query the default enabledness of collision detection between collision of the specified layers.

Examples

```
print( Scene.GetLayerCollisionDefault( CollisionLayer.User1, CollisionLayer.FurniturePlacement ) )
```

See Also

- Scene.SetLayerCollisionDefault

Scene.GetListenerObjects

Brief

Returns an array of scene objects that are listening for data.

Definition

```
Object[] Scene.GetListenerObjects( number startIndex = 1, number maxCount = -1 )
```

Arguments

startIndex - The position to start from. Previous indices are not added to the results.maxCount - The maximum size the results array can be.

Return Values

A table of Object instances.

Description

Returns an array of scene objects that are listening for data.

Examples

```
sceneListenerObjects = Scene.GetListenerObjects()
t = { letters={ "a", "b", "c" }, type="letter_message" }
for i, sceneListenerObject in ipairs( sceneListenerObjects ) do
    result1, result2, result3 = Object.SendListenerData( sceneListenerObject, t )
end
```

See Also

None

Scene.GetMaxSaveDataSize

Brief

Indicates the maximum size in bytes of the available save data storage area.

Definition

```
number Scene.GetMaxSaveDataSize()
```

Arguments

None

Return Values

The maximum size in bytes of the save data storage area.

Description

This function will return the maximum size in bytes of the available save data storage area. This number will never decrease. The number is valid only if the scene has save data access.

Examples

```
local access = Scene.GetSaveDataPermission()
if access == SaveDataPermission.ReadWrite and not Scene.IsSaveDataBusy() then
    mc = MemoryContainer.Create( Scene.GetMaxSaveDataSize() )
    -- modify container here
    Scene.SaveData( mc )
end
```

See Also

None

Scene.GetSaveDataPermission

Brief

Obtain the level of save data access available to the current user in this scene.

Definition

```
SaveDataPermission Scene.GetSaveDataPermission()
```

Arguments

None

Return Values

A permission level (see details).

Description

Some scenes (currently personal spaces and clubhouses) provide the ability for a script to save data. Public scenes will not provide this ability. Only the owner or club leader is currently able to read or write the save data, so the ReadOnly permission level below is not yet available. This function will return the save data access level for the current user in the current scene, from the following values:
SaveDataPermission.None
SaveDataPermission.ReadOnly
SaveDataPermission.ReadWrite
For convenience these values can also be compared in the expected order (see example). Note that the save data system may also temporarily be busy, even if the user has permission. This can be checked with the IsSaveDataBusy function.

Examples

```
local access = Scene.GetSaveDataPermission()
if (access > SaveDataPermission.None and not Scene.IsSaveDataBusy()) then
    Scene.LoadData(memContainer)
end
```

See Also

- Scene.IsSaveDataBusy

Scene.GetStartupInstanceParam

Brief

Returns the Instance parameter that was passed to the scene when it was initialised

Definition

```
number Scene.GetStartupInstanceParam()
```

Arguments

None.

Return Values

number - The Instance parameter that the scene was started with

Description

When the user is relocated to a new scene (either by using a relocate region placed in another scene, or by the LocalPlayer.Relocate function), an optional 'instance' parameter can be passed. The instance parameter is used to create different instances of the scene, and can then be queried by the destination scene's scene script to change the behaviour in this new instance of the scene. For example, you might have a 'lobby' scene and a 'game-room' scene connected to it. When the user chooses to move from the lobby to the game-room, the lobby's script might decide (for instance), to send all European users to game-room instance 1, and all US users to game-room instance 2. European users would all appear in one set of instances of the game-room, US users would all appear in another set of instances of the game-room. The game-room script could then use this function to determine if the current instance is a European or US one and change behaviour appropriately.

Examples

```
param = Scene.GetStartupInstanceParam()
```

See Also

Scene.GetStartupValueParam

Brief

Returns the value parameter that was passed to the scene when it was initialised

Definition

```
number Scene.GetStartupValueParam()
```

Arguments

None.

Return Values

number - The value parameter that the scene was started with

Description

The value parameter is usually derived from relocation regions in the scene editor. e.g. You might have one scene that has 2 doorways leading to copies of the same scene When placing the relocation region for the doorways, you add the name of the scene the doorways lead to and (optionally) a value and text parameter. Hence, the first doorway could lead to space B with value parameter 1, and the second doorway could lead to space B with value parameter 2. The scene script in space B could then use this function to, for instance, set different screen content depending on which doorway the user came through. Note that users entering the same scene will see other users in that scene, even though they might have entered with a different value parameter To create separate instances of the scene controlled by parameter, see Scene.GetStartupInstanceParam()

Examples

```
param = Scene.GetStartupValueParam()
```

See Also

Scene.GetVirtualObject

Brief

Returns the current scene's virtual Object and nil if not present.

Definition

```
Object Scene.GetVirtualObject()
```

Arguments

None

Return Values

the current scene's virtual Object and nil if not present.

Description

Returns the current scene's virtual Object and nil if not present.

Examples

```
sceneObject = Scene.GetVirtualObject()
t = { letters={ "a", "b", "c" }, type="letter_message" }
results = Object.SendListenerData( sceneObject, t )
```

See Also

None

Scene.IsLoading

Brief

Indicates if current scene is currently loading.

Definition

```
boolean Scene.IsLoading()
```

Arguments

None

Return Values

True if a scene is currently loading or false if fully loaded.

Description

A scene is considered loaded once the scene itself and all objects marked as load before spawn are loaded, and the local player is spawned. Object and scene scripts will start to run before a scene is fully loaded, this is by design as it allows background loading to continue where

necessary. If a script wishes to disable certain functionality while the scene is still loading, this function can be used to identify this state. Note that scene objects not marked as load before spawn will continue to load after this function returns true; to detect this state use Scene.IsLoadingObjects.

Examples

```
if (Scene.IsLoading()) then
    g_canStartGame = false          -- disallow playing a game until scene is loaded
end
```

See Also

- Scene.IsLoadingObjects

Scene.IsLoadingObjects

Brief

Indicates if current scene is currently loading scene objects.

Definition

```
boolean Scene.IsLoadingObjects()
```

Arguments

None

Return Values

True if a scene is currently loading scene objects or false if all scene objects are loaded.

Description

A scene is considered loaded once the scene itself and all objects marked as load before spawn are loaded, and the local player is spawned. Scene objects not marked as load before spawn will however continue to be downloaded after this point. This function will return true while there are still scene objects downloading and loading in the scene, and will only return false when all scene objects are loaded and active. If you only wish to detect when the scene and local player and all load before spawn objects are loaded, use Scene.IsLoading.

Examples

```
if (g_isLoading and not Scene.IsLoadingObjects()) then
    g_isLoading = false
    SendListenerDataToAllObjects()      -- user function
end
```

See Also

- Scene.IsLoading

Scene.IsSaveDataBusy

Brief

Returns true if the save data system is currently busy indicating a save data request would fail.

Definition

```
boolean Scene.IsSaveDataBusy()
```

Arguments

None

Return Values

true if the save data system is busy and false otherwise.

Description

This function is only useful if the user has save data access (see GetSaveDataPermission). If this function returns true, any attempt to use the save data system will cause a Lua error.

Examples

```
local access = Scene.GetSaveDataPermission()
if access == SaveDataPermission.ReadWrite and not Scene.IsSaveDataBusy() then
    mc = MemoryContainer.Create( Scene.GetMaxSaveDataSize() )
    -- modify container here
    Scene.SaveData( mc )
end
```

See Also

- Scene.GetSaveDataPermission

Scene.LoadData

Brief

Copies the scene's save data (if available) into the supplied memory container.

Definition

```
void Scene.LoadData( MemoryContainer memContainer, int size = Scene.GetMaxSaveDataSize( ), int offset
= 0 )
```

Arguments

memContainer - The memory container to load into.size - Number of bytes to obtain.offset - Offset in bytes into the memory container at which to store the data.

Return Values

None.

Description

All scenes which allow saving (clubhouses and personal space), have an allocated save data area which allow embedded objects and scenescripts to save data about the scene. Note that nothing should rely on specific data being present in the save data (as the user may delete their Home save data at any time) and thus actives should be able to recognise and handle the case where no save data is present. When a scene is accessed for the first time, all of its save data is set to zero.

Examples

```
local access = Scene.GetSaveDataPermission()
if access > SaveDataPermission.None and not Scene.IsSaveDataBusy() then
    local memContainer = MemoryContainer.Create( Scene.GetMaxSaveDataSize( ) )
    Scene.LoadData(memContainer)
    if (memContainer:GetUint16(0) == 48879) then        -- 0xBEEF
        print('Save data is valid: found the beef')
        g_mySize = memContainer:GetFloat32(4)
    end
end
```

See Also

- Scene.SaveData
- Scene.GetSaveDataPermission
- Scene.IsSaveDataBusy

Scene.LockPadControls

Brief

Locks the pad controls.

Definition

```
Scene.LockPadControls(boolean val)
```

Arguments

val - true or false to lock or unlock the pad controls respectively.

Return Values

Description

Locks the input pad controls, bar the SELECT button (for the Safe Screen). The developer is advised to use this with care, and to interact with the user appropriately during the time the controls are locked. If it's not obvious why the controls are locked, the user should be informed.

Examples

```
Scene.LockPadControls(true)
```

See Also

None

Scene.OverrideAvatarCloseUpCamAspect

Brief

Set the aspect for the standard close-up camera that follows the Avatar.

Definition

```
Scene.OverrideAvatarCloseUpCamAspect(boolean isOverride, number value)
```

Arguments

isOverride - whether or not to override the default setting.value - the value to set.

Return Values

None.

Description

Examples

```
Scene.OverrideAvatarCloseUpCamAspect(true, 16.0 / 9.0)
```

```
Scene.OverrideAvatarCloseUpCamAspect(false)
```

See Also

- Scene.OverrideAvatarCloseUpCamFov
- Scene.OverrideAvatarCloseUpCamZNear
- Scene.OverrideAvatarStandardCamAspect
- Scene.OverrideAvatarStandardCamZNear
- Scene.OverrideAvatarStandardCamZFar

Scene.OverrideAvatarCloseUpCamFov

Brief

Set the Field Of View for the standard close-up camera that follows the Avatar.

Definition

```
Scene.OverrideAvatarCloseUpCamFov(boolean isOverride, number value)
```

Arguments

isOverride - whether or not to override the default setting.value - the value to set.

Return Values

None.

Description

Examples

```
Scene.OverrideAvatarCloseUpCamFov(true, 60)
```

```
Scene.OverrideAvatarCloseUpCamFov(false)
```

See Also

- Scene.OverrideAvatarCloseUpCamAspect

- Scene.OverrideAvatarCloseUpCamZNear
- Scene.OverrideAvatarStandardCamAspect
- Scene.OverrideAvatarStandardCamZNear
- Scene.OverrideAvatarStandardCamZFar

Scene.OverrideAvatarCloseUpCamZFar

Brief

Set the far clipping for the standard close-up camera that follows the Avatar.

Definition

```
Scene.OverrideAvatarCloseUpCamZFar(boolean isOverride, number value)
```

Arguments

isOverride - whether or not to override the default setting.value - the value to set.

Return Values

None.

Description

Examples

```
Scene.OverrideAvatarCloseUpCamZFar(true, 1000)
```

```
Scene.OverrideAvatarCloseUpCamZFar(false)
```

See Also

- Scene.OverrideAvatarCloseUpCamFov
- Scene.OverrideAvatarCloseUpCamAspect
- Scene.OverrideAvatarStandardCamAspect
- Scene.OverrideAvatarStandardCamZNear
- Scene.OverrideAvatarStandardCamZFar

Scene.OverrideAvatarCloseUpCamZNear

Brief

Set the near clipping for the standard close-up camera that follows the Avatar.

Definition

```
Scene.OverrideAvatarCloseUpCamZNear(boolean isOverride, number value)
```

Arguments

isOverride - whether or not to override the default setting.value - the value to set.

Return Values

None.

Description

Examples

```
Scene.OverrideAvatarCloseUpCamZNear(true, 0.1)
```

```
Scene.OverrideAvatarCloseUpCamZNear(false)
```

See Also

- Scene.OverrideAvatarCloseUpCamFov
- Scene.OverrideAvatarCloseUpCamAspect
- Scene.OverrideAvatarStandardCamAspect
- Scene.OverrideAvatarStandardCamZNear
- Scene.OverrideAvatarStandardCamZFar

Scene.OverrideAvatarStandardCamAspect

Brief

Set the aspect for the standard close-up camera that follows the Avatar.

Definition

```
Scene.OverrideAvatarStandardCamAspect(boolean isOverride, number value)
```

Arguments

isOverride - whether or not to override the default setting.value - the value to set.

Return Values

None.

Description

Examples

```
Scene.OverrideAvatarStandardCamAspect(true, 16.0 / 9.0)
```

```
Scene.OverrideAvatarStandardCamAspect(false)
```

See Also

- Scene.OverrideAvatarStandardCamFov
- Scene.OverrideAvatarStandardCamZNear

- Scene.OverrideAvatarCloseUpCamAspect
- Scene.OverrideAvatarCloseUpCamZNear
- Scene.OverrideAvatarCloseUpCamZFar

Scene.OverrideAvatarStandardCamFov

Brief

Set the Field Of View for the standard close-up camera that follows the Avatar.

Definition

```
Scene.OverrideAvatarStandardCamFov(boolean isOverride, number value)
```

Arguments

isOverride - whether or not to override the default setting.value - the value to set.

Return Values

None.

Description

Examples

```
Scene.OverrideAvatarStandardCamFov(true, 60)
```

```
Scene.OverrideAvatarStandardCamFov(false)
```

See Also

- Scene.OverrideAvatarStandardCamAspect
- Scene.OverrideAvatarStandardCamZNear
- Scene.OverrideAvatarCloseUpCamAspect
- Scene.OverrideAvatarCloseUpCamZNear
- Scene.OverrideAvatarCloseUpCamZFar

Scene.OverrideAvatarStandardCamZFar

Brief

Set the far clipping for the standard close-up camera that follows the Avatar.

Definition

```
Scene.OverrideAvatarStandardCamZFar(boolean isOverride, number value)
```

Arguments

isOverride - whether or not to override the default setting.value - the value to set.

Return Values

None.

Description

Examples

```
Scene.OverrideAvatarStandardCamZFar(true, 1000)
```

```
Scene.OverrideAvatarStandardCamZFar(false)
```

See Also

- Scene.OverrideAvatarStandardCamFov
- Scene.OverrideAvatarStandardCamAspect
- Scene.OverrideAvatarCloseUpCamAspect
- Scene.OverrideAvatarCloseUpCamZNear
- Scene.OverrideAvatarCloseUpCamZFar

Scene.OverrideAvatarStandardCamZNear

Brief

Set the Far clipping for the standard close-up camera that follows the Avatar.

Definition

```
Scene.OverrideAvatarStandardCamZNear(boolean isOverride, number value)
```

Arguments

isOverride - whether or not to override the default setting.value - the value to set.

Return Values

None.

Description

Examples

```
Scene.OverrideAvatarStandardCamZNear(true, 0.1)
```

```
Scene.OverrideAvatarStandardCamZNear(false)
```

See Also

- Scene.OverrideAvatarStandardCamFov
- Scene.OverrideAvatarStandardCamAspect
- Scene.OverrideAvatarCloseUpCamAspect

- Scene.OverrideAvatarCloseUpCamZNear
- Scene.OverrideAvatarCloseUpCamZFar

Scene.ResetAvatarCamOverrides

Brief

Resets all previously set Avatar Camera overrides.

Definition

```
Scene.ResetAvatarCamOverrides()
```

Arguments

None

Return Values

None.

Description

Examples

```
Scene.ResetAvatarCamOverrides()
```

See Also

- Scene.OverrideAvatarStandardCamFov
- Scene.OverrideAvatarStandardCamAspect
- Scene.OverrideAvatarStandardCamZNear
- Scene.OverrideAvatarCloseUpCamAspect
- Scene.OverrideAvatarCloseUpCamZNear
- Scene.OverrideAvatarCloseUpCamZFar

Scene.SaveData

Brief

Copies the data from the supplied memory container into the scene's save data area and requests it to save.

Definition

```
Scene.SaveData( MemoryContainer memContainer, number size = Scene.GetMaxSaveDataSize( ), int offset =
0 )
```

Arguments

memContainer - The memory container to read data from.size - Number of bytes to obtain.offset - Offset in bytes into the memory container at which to start reading the data to save.

Return Values

None.

Description

All scenes which allow saving (clubhouses and personal spaces), have an allocated save data area which allow embedded objects and scene scripts to save data about the scene. Note that nothing should rely on specific data being present in the save data (as the user may delete their Home save data at any time) and thus a script should be able to recognise and handle the case where no or invalid save data is present. When a scene is accessed for the first time, all of its save data is set to zero. Note: If size is not specified and your memory container is bigger than the maximum save data size an error will occur to catch accidental data truncation.

Examples

```
local access = Scene.GetSaveDataPermission()
if (access == SaveDataPermission.ReadWrite and not Scene.IsSaveDataBusy()) then
    local memContainer = MemoryContainer.Create( 8 )
    memContainer:SetUint16(0, 48879)        -- 0xBEEF, tags our data as valid
    memContainer:SetFloat32(4, mySize)      -- save our current size so we can keep growing next time
    Scene.SaveData(memContainer, 8)
end
```

See Also

- Scene.LoadData
- Scene.GetSaveDataPermission
- Scene.IsSaveDataBusy

Scene.SetGraphicsEngineParam

Brief

Set the value of a parameter in the graphics engine.

Definition

```
Scene.SetGraphicsEngineParam(paramName, value)
```

Arguments

paramName - a valid graphics engine param name. Valid values are:
GraphicsEngine.ShadowEnable boolean - enable or disable shadows, defaults to true)
GraphiceEngine.ShadowQuality set the sun shadow quality. Should be one of:
ShadowQuality.Low
ShadowQuality.Medium
ShadowQuality.High
GraphicsEngine.SunlightColor Vector4 - set the color/brightness of the sun, w component is ignored. Each channel range is 0=>3
GraphicsEngine.SunlightDir Vector4 - set the direction of the sun. The value does not need to be normalized
GraphicsEngine.ColorBalance Vector4 - set the post-process color balance, components should be in the range -1=>1, defaults to (0,0,0)
GraphicsEngine.Saturation scalar - set the post-process saturation, should be in the range -2=>2, defaults to 1
GraphicsEngine.Contrast scalar - set the post-process contrast, should be in the range -1=>1, defaults to 0
GraphicsEngine.ShadowPlaneDistance scalar - distance to the first shadow plane in metres, defaults to 2.0
GraphicsEngine.ShadowPlaneMultiplier scalar - value to muliply each shadow plane distance by to get the distance to the next plane, defaults to 3.0 and must be greater than 1.0
GraphicsEngine.ShadowPlaneCount scalar - number of shadow planes to render, should be in the range 0=>4, defaults to 4
GraphicsEngine.ShadowDarkness scalar - set the darkness of indoor shadows (has no effect in sunlit scenes), should be in the range 0=>1, defaults to 0.2
GraphicsEngine.AlphaSortEnable boolean - enable or disable sorting of semitransparent meshes. for scenes exported with HDK1.40 onwards, this defaults to true.
GraphicsEngine.LightProbeTint Vector4 - set a global tint colour for the light probes. defaults to (1,1,1)
GraphicsEngine.GlowType set the type of glow. Should be one of:
GlowType.Disabled
GlowType.Small
GlowType.Default
GlowType.Large

GlowType.ExtraLarge
GlowType.Horizontal
GlowType.Vertical
GraphicsEngine.GlowColor Vector4 - set the tint colour for the glow
GraphicsEngine.GlowBlendMode set the blend type for the glow, should be one of:
GlowBlend.Default
GlowBlend.Add - add the glow buffer to the framebuffer
GlowBlend.Squared - square the glow buffer and add it to the framebuffer
GraphicsEngine.EnvironmentMap Set the current environment map for dynamic objects.
The argument should be the name of a texture resource, or the resource itselfvalue - the value to set.

Return Values

None.

Description

Examples

```
Scene.SetGraphicsEngineParam(GraphicsEngine.SunlightColor, Vector4.Create(2,2,0)) -- set the sun color
to bright yellow
Scene.SetGraphicsEngineParam(GraphicsEngine.ShadowEnable, false) -- disable shadows in this scene
```

See Also

None

Scene.SetLayerCollisionDefault

Brief

Set the default enabledness of collision detection between collision of the specified layers.

Definition

```
Scene.SetLayerCollisionDefault( enum layerA, enum layerB, boolean enabled )
```

Arguments

layerA - The first layer; must be a "user layer".layerB - The second layer.enabled - true to enable collision detection by default, false otherwise.

Return Values

None.

Description

Set the default enabledness of collision detection between collision of the specified layers.

Examples

```
Scene.SetLayerCollisionDefault( CollisionLayer.User1, CollisionLayer.FurniturePlacement, false )
```

See Also

- Scene.GetLayerCollisionDefault

Scene.SetLightProbeData

Brief

Set the light probe data for the scene

Definition

```
Scene.SetLightProbeData(string resourcename, number slotIndex, bool updateSun=false)
Scene.SetLightProbeData(Resource resource, number slotIndex, bool updateSun=false)
```

Arguments

resource - The probe data file resource, or resource name, or nil to clear the slot
This resource can not be released whilst it is in use on a slot. Set the slot to another resource or nil before
attempting to release it.slotIndex - The light probe volume index to set. There are 9 slots (numbered 0-8) available.updateSun - If this is true, update the sun parameters based on the values from the given probe data

Return Values

none

Description

Sets the light probe volume at the given index to the given data. If this volume overlaps an existing volume, then the one with the lowest slot number will have precedence. Note, there is no way to query the light probe slots once assigned.

Examples

```
-- load in new scene geometry --
entity = Entity.Create()
entity:SetModel( "my_new_scene_geometry.mdl" )
-- load corresponding lighting data --
Scene.SetLightProbeData( "mylighting.probe", 1 )
-- clear old lighting --
Scene.SetLightProbeData( nil, 0 )
```

See Also

- Scene.SetLightProbeTransform

Scene.SetLightProbeTransform

Brief

Set the transform for the light probe volume in the given slot

Definition

```
Scene.SetLightProbeTransform(Matrix44 transform, number slotIndex)
```

Arguments

transform - The transform to apply to the light probe volume in the given slotslotIndex - The light probe volume index to set. There are 9 slots (numbered 0-8) available.

Return Values

none

Description

Sets the transform for the light probe volume at the given index

Examples

```
-- load in new scene geometry --
entity = Entity.Create()
entity:SetModel( "my_new_scene_geometry.mdl" )
-- load corresponding lighting data --
Scene.SetLightProbeData( "mylighting.probe", 1 )
-- reposition model and probes --
pos = Vector4.Create(10,10,0)
entity:SetPosition( pos )
Scene.SetLightProbeTransform( Matrix44.Create( Quaternion.Create(), pos ), 1 )
```

See Also

- Scene.SetLightProbeData

SceneDownload

SceneDownload.CancelDownload

Brief

Cancels the specified scripted scene download.

Definition

```
boolean SceneDownload.CancelDownload( string sceneName )
```

Arguments

None

Return Values

True if download was cancelled, false otherwise.

Description

Only scene downloads triggered by script can be cancelled with this function.

Examples

```
if (SceneDownload.CancelDownload("myscene")) then
    print("Download cancelled")
end
```

See Also

- SceneDownload.StartDownload

SceneDownload.GetCurrentDownload

Brief

Returns the scripted scene download currently in progress (or nil if none is active).

Definition

```
string SceneDownload.GetCurrentDownload()
```

Arguments

None

Return Values

The current scene download, or nil if none is activate.

Description

Returns the scene name of the current scene download if one is active.

Examples

```
local scene = SceneDownload.GetCurrentDownload()
if (scene) then
    print("Downloading " .. scene .. "...")
end
```

See Also

- SceneDownload.StartDownload

SceneDownload.GetDownloadProgress

Brief

Gets the progress value for the download of the requested scene.

Definition

```
number SceneDownload.GetDownloadProgress( string sceneName )
```

Arguments

None

Return Values

A value between 0 (no data received) and 1 (all data received) indicating the progress of the download.

Description

Retrieves the progress proportion of the download of a specified scene.

Examples

```
local progress = SceneDownload.GetDownloadProgress("myscene")
RenderProgressBar(progress)     -- user function
```

See Also

- SceneDownload.StartDownload

SceneDownload.GetDownloadSize

Brief

Gets the size of the download for the specified scene name, in bytes.

Definition

```
number SceneDownload.GetDownloadSize( string sceneName )
```

Arguments

None

Return Values

The size in bytes of the specified scene download.

Description

Gets the size of the download for a specified scene. The download itself is not triggered.

Examples

```
local size = SceneDownload.GetDownloadSize("myscene")
```

See Also

None

SceneDownload.QuerySceneCacheStatus

Brief

Queries whether the specified scene is currently cached and up-to-date.

Definition

```
SceneDownload.QuerySceneCacheStatus( string sceneName )
```

Arguments

None

Return Values

One of the following result values:
CacheCheckResult.Downloading - scene is downloading
CacheCheckResult.InCache - scene is cached
CacheCheckResult.NeedsDownload - scene is out of date and needs re-downloading
CacheCheckResult.AccessRestricted - scene has failed age restriction or failed to download

Description

Queries whether the specified scene is currently cached and up-to-date.

Examples

```
if (SceneDownload.QuerySceneCacheStatus("myscene") == CacheCheckResult.InCache) then
    print("Scene is cached!")
end
```

See Also

- SceneDownload.StartDownload

SceneDownload.StartDownload

Brief

Starts a background download of the specified scene (if none is active).

Definition

```
boolean SceneDownload.StartDownload( string sceneName )
```

Arguments

None

Return Values

True if the scene download was started successfully, false otherwise.

Description

Triggers a background download of the named scene. Any download that is in progress will be cancelled.

Examples

```
if (SceneDownload.StartDownload("myscene")) then
    print("Scene download triggered!")
end
```

See Also

- SceneDownload.GetCurrentDownload

SceneInfo

SceneInfo.GetSceneId

Brief

Returns the sceneId of the current scene.

Definition

```
string SceneInfo.GetSceneId()
```

Arguments

None

Return Values

The sceneId.

Description

Returns the sceneId of the current scene.

Examples

```
print("Current scene ID:", SceneInfo.GetSceneId())
```

See Also

- SceneInfo.GetSceneName
- SceneInfo.GetUniqueInstanceId

SceneInfo.GetSceneName

Brief

Returns the localized name of the current scene.

Definition

```
string SceneInfo.GetSceneName()
```

Arguments

None

Return Values

The localized name of the scene.

Description

Returns the localized name of the current scene.

Examples

```
print("Entering scene:", SceneInfo.GetSceneName())
```

See Also

- SceneInfo.GetSceneId
- SceneInfo.GetUniqueInstanceId

SceneInfo.GetUniqueInstanceId

Brief

Return a unique instance ID for the current scene instance.

Definition

```
string SceneInfo.GetUniqueInstanceId()
```

Arguments

None

Return Values

The instance ID.

Description

This function will construct and return a unique ID string for the current scene instance. This value will be unique for different instances of the same scene. No assumptions should be made about the format of the instance ID.

Examples

```
print("Scene instance ID:", SceneInfo.GetUniqueInstanceId())
```

See Also

- SceneInfo.GetSceneId
- SceneInfo.GetSceneName

SceneTransition

SceneTransition.ExitScript

Brief

Terminates the SceneTransition object.

Definition

```
SceneTransition.ExitScript()
```

Arguments

None

Return Values

None.

Description

This function must be called when the scene transition object has finished its execution to release the object and hand control over to the user.

Examples

```
if (SceneTransition.IsSceneChangeComplete()) then
    SceneTransition.ExitScript()
end
```

See Also

None

SceneTransition.GetDestinationScene

Brief

Returns the name of the destination scene for a scene transition.

Definition

```
string SceneTransition.GetDestinationScene()
```

Arguments

None

Return Values

The scene name.

Description

Returns the name of the destination scene for a scene transition.

Examples

```
local scene = SceneTransition.GetDestinationScene()
```

See Also

- SceneTransition.GetOriginScene

SceneTransition.GetObject

Brief

Returns the SceneTransition object (if it is running).

Definition

```
Object SceneTransition.GetObject()
```

Arguments

None

Return Values

SceneTransition object if it is running, or nil otherwise.

Description

Returns the SceneTransition object if a relocation process is occurring.

Examples

```
local object = SceneTransition.GetObject()
```

See Also

None

SceneTransition.GetOriginScene

Brief

Returns the name of the scene that started the scene transition.

Definition

```
string SceneTransition.GetOriginScene()
```

Arguments

None

Return Values

The scene name.

Description

Returns the name of the scene that started the scene transition.

Examples

```
local scene = SceneTransition.GetOriginScene()
```

See Also

- SceneTransition.GetDestinationScene

SceneTransition.GetSceneChangeProgress

Brief

Returns a value indicating the progress of the current relocation from 0.0 to 1.0.

Definition

```
number SceneTransition.GetSceneChangeProgress()
```

Arguments

None

Return Values

A value between 0 and 1 indicating the transition progress.

Description

Returns a value indicating the progress of the current relocation from 0.0 to 1.0.

Examples

```
local progress = SceneTransition.GetSceneChangeProgress()
RenderProgressBar(progress)         -- user function
```

See Also

None

SceneTransition.IsSceneChangeComplete

Brief

Indicates when the scene change is fully complete and the new scene is ready to interact with.

Definition

```
boolean SceneTransition.IsSceneChangeComplete()
```

Arguments

None

Return Values

True if the scene change is complete, or false if still in progress.

Description

If the scene change has completed and the new scene is ready to interact with, this function will return true.

Examples

```
if (SceneTransition.IsSceneChangeComplete()) then
    print("Scene ready!")
end
```

See Also

None

SceneTransition.StartSceneChange

Brief

Begins the scene relocation process.

Definition

```
boolean SceneTransition.StartSceneChange()
```

Arguments

None

Return Values

True if scene change started successfully, false otherwise.

Description

Activates the scene relocation, removing the user from the current scene and starting the process of loading and moving to the new one.

Examples

```
if (SceneTransition.StartSceneChange()) then
    print("Scene change started!")
end
```

See Also

None

Screen

Screen.AreSceneSoundsMuted

Brief

Return if scene sound is muted while in fullscreen

Definition

```
boolean Screen.AreSceneSoundsMuted(Screen screen)
```

Arguments

screen

Return Values

true if scene sound is muted while in fullscreen

Description

Return if scene sound is muted while in fullscreen

Examples

```
local screen = Scene.FindScreen( "video1" )
screen.AreSceneSoundsMuted()
```

See Also

- Scene.FindScreen

Screen.Create

Brief

Creates a new screen.

Definition

```
Screen Screen.Create( Vector4 topLeft, Vector4 bottomRight, Vector4 bottomLeft, string type, string
source = nil )
```

Arguments

topLeft - World space coordinate of top left hand corner of the screen.bottomRight - World space coordinate of bottom right hand corner of the screen.bottomLeft - World space coordinate of bottom left hand corner of the screen.type - String denoting the type of screen to be created. Can be one of the following:
"HSML"
"IMAGE"
"LUA"
"MEDIARSS"
"VIDEO"
source - Source for the data to be displayed on the screen (a URL or file path).

Return Values

A new Screen instance.

Description

Creates a new screen. The screen type can be one of the following: "HSML" - for displaying 'Home Screen Markup Language' pages. "IMAGE" - for displaying a DDS, JPG or PNG file. "LUA" - for a screen that can have a Renderer object attached. These will be protected by default. "MEDIARSS" - for displaying media syndicated via an RSS feed. "VIDEO" - for displaying video content.

Examples

```
-- Create a screen for submitting graphical primitives from Lua
spriteScreen = Screen.Create( topLeft, bottomRight, bottomLeft, "LUA" )
renderer = Renderer.Create()
Renderer.SetTarget(renderer, spriteScreen)
Renderer.DrawSprite(renderer, sprite)

-- Create a video screen
url = "http://www.example.com/video.mp4"
vidScreen = Screen.Create( topLeft, bottomRight, bottomLeft, "VIDEO", url )
```

See Also

- Renderer.SetTarget
- Screen.SetProtected

Screen.EnableBackground

Brief

Enables the default background for IMAGE screen types.

Definition

```
Screen.EnableBackground( Screen screen, boolean disable )
```

Arguments

screen - The screen to toggle the background for.disable - true to enable the default background, false to disable it (default is enabled).

Return Values

None.

Description

Enables or disables the the default black background for IMAGE screen types.

Examples

```
local screen = Scene.FindScreen( "Poster" )
screen.EnableBackground( false )
```

See Also

- Scene.FindScreen

Screen.GetCenter

Brief

Returns a vector holding the central position of the screen.

Definition

```
Vector4 Screen.GetCenter( Screen screen )
Screen.GetCenter( Screen screen, Vector4 outPos )
```

Arguments

screen - The screen to query.outPos - (out) The vector to receive the position.

Return Values

A vector holding the central position of the screen.

Description

Returns a vector holding the central position of the screen. This vector is in world space.

Examples

```
screen = Scene.FindScreen( "MyVideoScreen" )
center = Screen.GetCenter( screen )
print( "The screen center point is " .. center )
```

See Also

- ArcadeGame.GetScreen
- Scene.FindScreen

- Screen.Create

Screen.GetContentCount

Brief

Returns the number of different content attached to the screen.

Definition

```
number Screen.GetContentCount( Screen screen )
```

Arguments

screen - The screen to query.

Return Values

The number of different content attached to the screen.

Description

Returns the number of different content attached to the screen.

Examples

```
-- Step to the next content source if it has been downloaded
screen = Scene.FindScreen( "MyVideoScreen" )

contentCount = Screen.GetContentCount( screen )
currentIndex = Screen.GetCurrentContentIndex( screen )

nextIndex = ( currentIndex + 1 ) % contentCount
if Screen.IsContentReady( screen, nextIndex ) then
    Screen.SetCurrentContentIndex( screen, nextIndex )
end
```

See Also

- Scene.FindScreen
- Screen.GetCurrentContentIndex
- Screen.IsContentReady
- Screen.SetCurrentContentIndex

Screen.GetCurrentContentIndex

Brief

Returns the current source index of the screen.

Definition

```
number Screen.GetCurrentContentIndex( Screen screen )
```

Arguments

screen - The screen to query.

Return Values

The current content index of the screen.

Description

Returns the current source index of the screen.

Examples

```
-- Step to the next content source if it has been downloaded
screen = Scene.FindScreen( "MyVideoScreen" )

contentCount = Screen.GetContentCount( screen )
currentIndex = Screen.GetCurrentContentIndex( screen )

nextIndex = ( currentIndex + 1 ) % contentCount
if Screen.IsContentReady( screen, nextIndex ) then
    Screen.SetCurrentContentIndex( screen, nextIndex )
end
```

See Also

- Scene.FindScreen
- Screen.GetContentCount
- Screen.IsContentReady
- Screen.SetCurrentContentIndex

Screen.GetCurrentTime

Brief

Returns the current time of the screen in seconds.

Definition

```
number Screen.GetCurrentTime( Screen screen )
```

Arguments

screen - The screen to query.

Return Values

The current time of the screen in seconds.

Description

This is currently only supported on video screens and media-rss screens currently playing video content. Note: If the content is not ready it will return 0. Use Screen.IsContentReady to check the status.

Examples

```
screen = Scene.FindScreen( "MyVideoScreen" )
if ( Screen.GetCurrentTime( screen ) >= 20 ) then
    state = STATE_CHEER
end
```

See Also

- Scene.FindScreen
- Screen.GetTotalTime

Screen.GetMatrix

Brief

Returns the world matrix for the screen.

Definition

```
Screen.GetMatrix( Screen screen, Matrix44 matrix )
```

Arguments

screen - The screen to query.matrix - A matrix to set.

Return Values

None.

Description

Returns the world matrix for the screen. An identity matrix is returned if the screen is invalid. Note that the world matrix will scale the resolution of the screen to the world-space size of the screen, and thus will typically include a large scale component, so do not assume that any row of the matrix will be unit length. If only the screen normal is required, use Screen.GetNormal.

Examples

```
local matrix = Matrix44.Create()
screen = Scene.FindScreen( "MyVideoScreen" )
Screen.GetMatrix( screen, matrix )
print( 'Screen translation is ', matrix.GetRow( 4 ) )
```

See Also

- ArcadeGame.GetScreen
- Scene.FindScreen
- Screen.Create
- Screen.GetNormal

Screen.GetNormal

Brief

Returns the normal for the screen.

Definition

```
Screen.GetNormal( Screen screen, Vector4 vector )
```

Arguments

screen - The screen to query.vector - A vector to set to the normal.

Return Values

None.

Description

Returns the normal for the screen, that is, the vector pointing in the facing direction of the screen. The normal will be unit length.

Examples

```
local vector = Vector4.Create()
screen = Scene.FindScreen( "MyVideoScreen" )
Screen.GetNormal( screen, vector )
print( "The screen normal is ", vector )
```

See Also

- ArcadeGame.GetScreen
- Scene.FindScreen
- Screen.Create

Screen.GetSize

Brief

Returns a vector holding the screen size.

Definition

```
Vector4 Screen.GetSize( Screen screen )
Screen.GetSize( Screen scree, Vector4 outSize )
```

Arguments

screen - The screen to query.outSize - (out) The vector to receive the size.

Return Values

A vector holding the size of the screen's content in pixels. Only X and Y vector components are valid.

Description

Returns a vector holding the size of the screen's content in pixels. The X component of the returned vector holds the screen's content width and the Y component holds the screen's content height.

Examples

```
screen = Scene.FindScreen( "MyVideoScreen" )
contentSize = Screen.GetSize( screen )
print( "The screen content size is " .. contentSize )
```

See Also

- ArcadeGame.GetScreen
- Scene.FindScreen
- Screen.Create

Screen.GetTotalTime

Brief

Returns the current time of the screen in seconds.

Definition

```
number Screen.GetTotalTime( Screen screen )
```

Arguments

screen - The screen to query.

Return Values

The total time of the screen in seconds.

Description

This returns the total duration of the currently playing video. This is currently only supported on video screens and media-rss screens playing video content. Note: If the content is not ready it will return 0. Use Screen.IsContentReady to check the status.

Examples

```
screen = Scene.FindScreen( "MyVideoScreen" )
if ( Screen.GetCurrentTime( screen ) / Screen.GetTotalTime( screen ) >= 0.5 ) then
    state = STATE_HALF_WAY
end
```

See Also

- Scene.FindScreen

Screen.GetTriggerPosition

Brief

Retrieves the trigger position for the screen.

Definition

```
Screen.GetTriggerPosition( Screen screen, Vector4 position )
```

Arguments

screen - Screen object to query.position - The current trigger position.

Return Values

None.

Description

Retrieves the current trigger position, which determines where the user can target this screen. If a custom trigger position has not been set this will return the same as Screen.GetCenter(). This will set a position of 0 if the screen is not valid.

Examples

```
local position = Vector4.Create()
screen = Scene.FindScreen( "MyVideoScreen" )
Screen.GetTriggerPosition( screen, position )
```

See Also

- Screen.SetTriggerPosition
- Screen.GetTriggerRadius
- Screen.Create
- Scene.FindScreen

Screen.GetTriggerRadius

Brief

Retrieves the trigger radius for the screen.

Definition

```
number Screen.GetTriggerRadius( Screen object )
```

Arguments

object - Screen object to query.

Return Values

range - The current trigger radius in meters.

Description

Retrieves the current trigger radius, which determines from how far away the user can target this screen. Note that 0 is considered as an infinite target radius.

Examples

```
range = Screen.GetTriggerRadius( object )
```

See Also

- Screen.SetTriggerRadius
- Screen.Create
- Scene.FindScreen

Screen.GetVolume

Brief

Returns the volume of the screen.

Definition

```
Number Screen.GetVolume( Screen screen )
```

Arguments

screen - The screen to query.

Return Values

A number from 0.0 to 1.0 indicating the current volume of the screen.

Description

Returns the current volume of the screen. A value of 0.0 will be returned if the screen is invalid.

Examples

```
screen = Scene.FindScreen( "MyVideoScreen" )
volume = Screen.Getvolume( screen )
print( "The screen volume size is " .. volume )
```

See Also

- Scene.FindScreen
- Screen.Create
- Screen.SetVolume

Screen.IsContentPlaying

Brief

Returns true if the content is playing and false otherwise.

Definition

```
boolean Screen.IsContentPlaying( Screen screen )
```

Arguments

screen - The screen to query.

Return Values

Returns true if the content is playing and false otherwise.

Description

This is currently only supported on video screens and media-rss screens currently playing video content. Note: If the content is not ready it will return false. Use Screen.IsContentReady to check the status.

Examples

```
if Screen.IsContentPlaying( screen ) == true then
    Screen.Pause( screen )
end
```

See Also

- Screen.Pause
- Screen.Resume

Screen.IsContentReady

Brief

Reports whether the content is ready to be displayed.

Definition

```
boolean Screen.IsContentReady( Screen screen, number contentIndex )
```

Arguments

screen - The screen to query.contentIndex - The source index to query. Ranges from 1 to the content count.

Return Values

True if the content is ready and false otherwise.

Description

Reports whether the content is ready to be displayed. This is only supported for video screens and media-rss screens while playing video content.

Examples

```
-- Step to the next content source if it has been downloaded
screen = Scene.FindScreen( "MyVideoScreen" )

contentCount = Screen.GetContentCount( screen )
currentIndex = Screen.GetCurrentContentIndex( screen )

nextIndex = ( currentIndex + 1 ) % contentCount
if Screen.IsContentReady( screen, nextIndex ) then
    Screen.SetCurrentContentIndex( screen, nextIndex )
end
```

See Also

- Scene.FindScreen
- Screen.GetContentCount
- Screen.GetCurrentContentIndex
- Screen.SetCurrentContentIndex

Screen.IsFogEnabled

Brief

Retrieve the fog enable state for a screen.

Definition

```
boolean Screen.IsFogEnabled(Screen screen)
```

Arguments

screen - the screen to access

Return Values

True if the screen is affected by the scene fog or false otherwise.

Description

Retrieve the fog enable state for a screen. If this value is true the screen will be affected by the fog settings in the scene.

Examples

```
fogEnabled = myScreen:IsFogEnabled();
```

See Also

- Screen.SetFogEnabled

Screen.IsHDFullScreenModeSet

Brief

Return if a video is in real full screen mode

Definition

```
boolean Screen.IsHDFullScreenModeSet(Screen screen)
```

Arguments

screen

Return Values

true if real full screen, false otherwise

Description

Return if a video is in real full screen mode

Examples

```
local screen = Scene.FindScreen( "video1" )
screen.IsHDFullScreenModeSet()
```

See Also

- Scene.FindScreen

Screen.IsInFullScreenMode

Brief

Returns true if the screen is in full screen mode and false otherwise.

Definition

```
boolean Screen.IsInFullScreenMode(Screen screen)
```

Arguments

Screen - The screen to query.

Return Values

True if the screen is in full screen mode and false otherwise.

Description

Examples

```
screen = Scene.FindScreen("MyVideoScreen")
isFullScreenMode = Screen.IsInFullScreenMode(screen)
```

See Also

None

Screen.IsProtected

Brief

Retrieve the protection state for a screen.

Definition

```
boolean Screen.IsProtected(Screen screen)
```

Arguments

screen - the screen to access

Return Values

True if the screen is protected or false otherwise.

Description

Retrieve the protection state for a screen. If this value is true the screen will not be rendered into any still or video captures performed by PlayStation(R) Home.

Examples

```
if (not myVidScreen:IsProtected()) then
    myVidScreen:SetProtected(true)
end
```

See Also

- Screen.SetProtected

Screen.IsVisible

Brief

Returns whether a screen is visible or not.

Definition

```
Boolean Screen.IsVisible( Screen object )
```

Arguments

object - Screen object to affect.isVisible - true or false (defaults to true).

Return Values

Whether the screen is visible or not.

Description

This function returns whether a screen has been tagged as visible or invisible.

Examples

```
isVisible = Screen.IsVisible( object )
```

See Also

- Screen.SetVisible

Screen.Pause

Brief

Pauses the currently playing screen.

Definition

```
Screen.Pause( Screen screen )
```

Arguments

screen - The screen to be paused.

Return Values

None.

Description

Pauses the currently playing screen.
Note: This is currently only supported for video, hsml and media-rss screens.

Examples

```
s = Scene.FindScreen( "MyVideoScreen" )
Screen.Pause( s )
```

See Also

- Screen.Resume

Screen.RestartStream

Brief

Restart the stream.

Definition

```
boolean Screen.RestartStream( Screen object )
```

Arguments

object - Screen object to restart.

Return Values

true if succeeded

Description

Restart the current stream.

Examples

```
boolReturn = Screen.RestartStream( object )
```

See Also

None

Screen.Resume

Brief

Resumes the currently paused screen.

Definition

```
Screen.Resume( Screen screen )
```

Arguments

screen - The screen to be resumed.

Return Values

None.

Description

Resumes the currently paused screen.
Note: This is currently only supported for video, hsml and media-rss screens.

Examples

```
s = Scene.FindScreen( "MyVideoScreen" )
Screen.Resume( s )
```

See Also

- Screen.Pause

Screen.SetChannel

Brief

Sets the content channel for the screen.

Definition

```
Screen.SetChannel( Screen screen, string channelName )
```

Arguments

screen - The screen to be updated.channelName - the name of the content channel to display on the screen.

Return Values

None.

Description

Home stores a list of media content 'channels' - e.g. one channel might be "SonyAdChannel1" - with each channel providing an indirection to the screen type and content. The content for each channel can be updated independently of the main package and scene data.

Examples

```
Screen.SetChannel( myScreen, "SonyAdChannel1" )
```

See Also

- Screen.SetContent
- Screen.SetType

Screen.SetContent

Brief

Sets a content source or sources on the screen.

Definition

```
Screen.SetContent( Screen screen, string[] contentList )
Screen.SetContent( Screen screen, string content, number contentIndex = 1 )
```

Arguments

screen - The screen to update.contentList - An array of strings specifying multiple content sources.content - A string for the content source.contentIndex - The index to set the content to. Ranges from 1 to the number of content sources. Defaults to 1.

Return Values

None.

Description

Sets a content source or sources on the screen. The source string will either be a URL or a path relative to the HDK's build root directory. Note that paths relative to the HDK's build root directory can only be used for testing/development purposes. Scripts uploaded to a content server must reference content served over HTTP. Also note that HomeDeveloper.self must be launched in online mode in order for online sources to be specified.

Examples

```
-- Example1: Testing a video served from the host PC's hard drive for debug
localSource = "localvideo/testvideo.mp4"
Debug.SetContent( myScreen1, localSource )

-- Example 2: Play back a video served from an HTTP server
onlineSource = "http://mycontentserver.com/onlinevideo.mp4"
Screen.SetContent( myScreen2, onlineSource )
```

See Also

- Scene.FindScreen

Screen.SetCurrentContentIndex

Brief

Sets the screen to the specified content index.

Definition

```
Screen.SetCurrentContentIndex( Screen screen, number contentIndex )
```

Arguments

screen - The screen to update.contentIndex - The index of the content to set the screen to. Ranges from 1 to the content count.

Return Values

None.

Description

Sets the screen to the specified content index.

Examples

```
-- Step to the next content source if it has been downloaded
screen = Scene.FindScreen( "MyVideoScreen" )

contentCount = Screen.GetContentCount( screen )
currentIndex = Screen.GetCurrentContentIndex( screen )

nextIndex = ( currentIndex + 1 ) % contentCount
if Screen.IsContentReady( screen, nextIndex ) then
    Screen.SetCurrentContentIndex( screen, nextIndex )
end
```

See Also

- Scene.FindScreen
- Screen.GetContentCount
- Screen.GetCurrentContentIndex
- Screen.IsContentReady

Screen.SetCurrentTime

Brief

Set the current time for the currently selected content.

Definition

```
Screen.SetCurrentTime(Screen screen, number time)
```

Arguments

screen - the screen to modifytime - the time (in seconds) to set

Return Values

Description

Set the current playback time for the currently selected content. For a video this will set the playback time to "time" seconds from the beginning of the video for example. This is currently only supported on video screens and media-rss screens playing video content. Note that setting the time will necessarily cause the video to rebuffer, so there will be a delay before the movie changes to the new time. This is worth bearing in mind if you are attempting to synchronize video playback between clients.

Examples

```
s = Screen.Create( 'video', 'http://www.example.com/video.mp4' )
s:Pause()
s:SetCurrentContentIndex( 3 )
s:SetCurrentTime( timeOffset )      -- timeOffset calculated to synchronise this user with the others
s:Resume()
```

See Also

- Screen.Pause
- Screen.Resume
- Screen.GetCurrentTime

Screen.SetData

Brief

Sets the data for an HSML screen.

Definition

```
Screen.SetData( Screen screen, <string|MemoryContainer|Resource> data, string contentRoot )
```

Arguments

screen - The screen to update.data - Different types depending on the type of the screen.contentRoot - Root path to which all content referenced by the HSML is relative to.

Return Values

Description

Currently only supported for HSML screens. For HSML screens the data can be either a string, a MemoryContainer containing BasicGenx output, or an XML Resource.

Examples

```
-- Example 1: Setting HSML content from a string
hsmlScreen = Scene.FindScreen( "HsmlScreen" )
data = '<XML><PAGE><RECT X="0" Y="0" W="1280" H="720" col="#FF00C0" /><TEXT X="430" Y="20"
col="#FFFFFF" size="4">'
data = data .. Person.GetName( LocalPlayer.GetPerson() )
data = data .. '</TEXT></PAGE></XML>'
Screen.SetData( hsmlScreen, data )

-- Example 2: Setting HSML content from a MemoryContainer
mem = MemoryContainer.Create( 4096 )
writer = BasicGenx.Create( mem )
BasicGenx.StartDocument( writer )
-- Use BasicGenx to write HSML XML data here...
BasicGenx.EndDocument( writer )
Screen.SetData( hsmlScreen, mem )

-- Example 3: Setting HSML content from a downloaded XML resource
url = "http://www.example.com/hsmlscreen.xml"
hsmlResource = Resource.Request( url, "xml" )
while Resource.IsLoading( hsmlResource ) do
    coroutine.yield()
end
if Resource.IsLoaded( hsmlResource ) then
    Screen.SetData( hsmlScreen, hsmlResource )
end
```

See Also

- BasicGenx.Create
- MemoryContainer.Create
- Resource.Request
- Scene.FindScreen

Screen.SetFogEnabled

Brief

Set whether this screen should be affected by fog (defaults to false)

Definition

```
Screen.SetFogEnabled(Screen screen, boolean enable)
```

Arguments

screen - the screen to modifyenable - true to enable fogging or false otherwise

Return Values

None.

Description

Set whether this screen should be affected by fog. If this flag is set to true, the screen will be affected by the fog in the scene, and will therefore need to be positioned in front of a solid backing in order to provide the z-value (screens themselves do not write to the depth buffer).

Examples

```
myScreen:SetFogEnabled( true )
```

See Also

- Screen.IsFogEnabled

Screen.SetHDFullScreenMode

Brief

Set a video full screen mode to real full screen.

Definition

```
Screen.SetHDFullScreenMode( Screen screen, boolean value )
```

Arguments

screen - The screen to toggle the background for.value - true to enable real full screen, false to disable it (default is disabled).

Return Values

None.

Description

Set a video full screen mode to real full screen.

Examples

```
local screen = Scene.FindScreen( "video1" )
screen.SetHDFullScreenMode( true )
```

See Also

- Scene.FindScreen

Screen.SetHighPriority

Brief

Set a video screen to high priority.

Definition

```
Screen.SetHighPriority( Screen screen, boolean value )
```

Arguments

screen - The screen to toggle the background for.value - true to enable high priority, false to disable it (default is disabled).

Return Values

None.

Description

Set a video screen to high priority.

Examples

```
local screen = Scene.FindScreen( "video1" )
screen.SetHighPriority( false )
```

See Also

- Scene.FindScreen

Screen.SetMuteSceneSounds

Brief

Set the sounds in the scene to mute if video in full screen.

Definition

```
Screen.SetMuteSceneSounds( Screen screen, boolean value )
```

Arguments

screenvalue - true to disable scene sound while in full screen, false to disable it (default is disabled).

Return Values

None.

Description

Set the sounds in the scene to mute if video in full screen.

Examples

```
local screen = Scene.FindScreen( "video1" )
screen.SetMuteSceneSounds( true )
```

See Also

- Scene.FindScreen

Screen.SetProtected

Brief

Enable or disable capture protection on a screen.

Definition

```
Screen.SetProtected(Screen screen, boolean protected)
```

Arguments

screen - the screen to modifyprotected - true to protect the screen or false otherwise

Return Values

None.

Description

Enable or disable capture of a screen's contents. If this flag is set to true, any capture process (screenshot or video) will cause the screen's content to not be rendered, thus protecting any copyrighted or sensitive material. Setting to false will unprotect the screen's content. "LUA" screens will default to protected, other screen types will default to unprotected.

Examples

```
-- Create a screen for submitting graphical primitives from Lua
vidScreen = Screen.Create( topLeft, bottomRight, bottomLeft, "VIDEO",
"http://www.example.com/CopyrightedVideo.mp4" )
vidScreen:SetProtected( true )
```

See Also

- Screen.IsProtected

Screen.SetTargetable

Brief

Sets the screen to be targetable or not.

Definition

```
Screen.SetTargetable( Screen screen, boolean value )
```

Arguments

screen - The screen to update.value - If true then the screen is targetable and if false it is not targetable.

Return Values

None.

Description

Sets the screen to be targetable or not. Screens created in Lua default to have targeting disabled. Screens that appear in the HDK's Scene Editor application default to have targeting enabled.

Examples

```
-- Example 1: This type of screen defaults to targetable
screen1 = Scene.FindScreen( "MyVideoScreen" )
Screen.SetTargetable( screen1, false )

-- Example 2: This type of screen defaults to not targetable
screen2 = Screen.Create( topLeft, bottomRight, bottomLeft, "LUA" )
Screen.SetTargetable( screen2, true )
```

See Also

- Scene.FindScreen
- Screen.Create

Screen.SetTriggerPosition

Brief

Changes the trigger position for the screen.

Definition

```
Screen.SetTriggerPosition( Screen object, Vector4 position = nil )
```

Arguments

object - Screen object to modify.position - The new trigger position. If nil is specified the trigger position will reset to default.

Return Values

None.

Description

Modifies the default trigger position to affect where the user can target this screen. Specifying nil will reset the trigger position to be based on Screen.GetCenter().

Examples

```
local position = Vector4.Create( 10, 0, 10 )
local screen = Scene.FindScreen( "MyVideoScreen" )
Screen.SetTriggerPosition( screen, position ) -- set a custom trigger position
Screen.SetTriggerPosition( screen, nil ) -- reset the position to default
```

See Also

- Screen.GetTriggerPosition
- Screen.SetTriggerRadius

- Screen.Create
- Scene.FindScreen

Screen.SetTriggerRadius

Brief

Changes the trigger radius for the screen.

Definition

```
Screen.SetTriggerRadius( Screen object, number range )
```

Arguments

object - Screen object to modify.range - The new trigger radius in meters. Must be positive or 0, 0 is considered infinite.

Return Values

None.

Description

Modifies the default trigger radius to affect from how far away the user can target this screen. Specifying 0 will make the screen targetable from any distance.

Examples

```
Screen.SetTriggerRadius( screen, 10 )
```

See Also

- Screen.GetTriggerRadius
- Screen.Create
- Scene.FindScreen

Screen.SetType

Brief

Sets the content type for the screen.

Definition

```
Screen.SetType( Screen screen, string type )
```

Arguments

screen - The screen to update.type - The new screen type. Can be one of the following:
"HSML"
"IMAGES"
"LUA"
"MEDIARSS"
"VIDEO"

Return Values

None.

Description

Sets the content type for the screen.

Examples

```
Screen.SetType( myScreen, "VIDEO" )
Screen.SetContent( myScreen, "http://www.example.com/myvideo.mp4" )
```

See Also

- Screen.SetContent

Screen.SetVisible

Brief

Set screen to be visible or invisible.

Definition

```
Screen.SetVisible( Screen object, boolean isVisible )
```

Arguments

object - Screen object to affect.isVisible - true to make the screen visible and false otherwise.

Return Values

None.

Description

Set screen to be visible or invisible. This will also mute videos and remove the ability for the screen to be targetable. Will error when applied to a geometry screen.

Examples

```
Screen.SetVisible( object, false )
```

See Also

- Screen.IsVisible

Screen.SetVolume

Brief

Sets the volume of the screen.

Definition

```
Screen.SetVolume( Screen screen, number volume )
```

Arguments

screen - The screen to query.volume - The volume to set, 0.0 to 1.0.

Return Values

None.

Description

Sets the volume of the screen. A value outside the 0.0 to 1.0 range will be capped.

Examples

```
screen = Scene.FindScreen( "MyVideoScreen" )
Screen.SetVolume( screen, 0.5 )
```

See Also

- Scene.FindScreen
- Screen.Create
- Screen.GetVolume

Screenshot

Screenshot.Capture

Brief

Start a screen capture.

Definition

```
boolean Screenshot.Capture(Screenshot shot,
                           boolean showScreens = false,
                           boolean showOsd = true,
                           boolean showLabels = false,
                           boolean showChat = false,
                           boolean showWatermark = true)
```

Arguments

shot - the screenshot object to useshowScreens - enable screen rendering for screenshotshowOsd - enable OSD rendering for screenshotshowLabels - show player labelsshowChat - show player chat bubblesshowWatermark - show PlayStation Home watermark on screenshot

Return Values

true if capture is successfully started, or false if an existing screenshot is still capturing

Description

Start a screen capture with the specified options. Note that a true return value does not indicate that the capture has finished, the script must wait until Screen.IsInProgress returns false. The following arguments must not be changed from their default values without permission from a PlayStation(R)Home Regional Manager: * showScreens * showLabels * showChat Any screens marked as protected will not be shown regardless of the value of showScreens. Any changes to screen, OSD or label rendering is temporary and only active for the duration of the screenshot. The developer should consult their regional SCE manager about the legal ramifications of enabling screen rendering in screenshots. Note that Lua screens are excepted from this: even if showScreens is false, Lua-created screens will be still be shown, as they are often required to maintain the look of scene.

Examples

```
shot = Screenshot.Create()
capturing = shot:Capture(true, true, false, false, true)    -- everything on except labels and chat.
if (not capturing) then
    print('Capture failed')
end
...
function OnUpdate()
    if (capturing and not shot:IsInProgress() and shot:Succeeded()) then
        MediaLibrary.ExportImage( shot, "Home Screenshot" )
    end
end
```

See Also

- Screenshot.Create
- Screenshot.IsInProgress
- Screenshot.Succeeded
- Screen.IsProtected

Screenshot.Create

Brief

Create a new screenshot object.

Definition

```
Screenshot Screenshot.Create()
```

Arguments

None

Return Values

A new screenshot object, or nil on failure

Description

Creates a new screenshot object, used to save an image of the current frame buffer contents. Only a limited number of screenshot objects may exist at once, if this becomes exhausted, nil is returned so you can either fail or keep retrying every frame until another script releases a Screenshot handle.

Examples

```
shot = Screenshot.Create()
```

See Also

- Screenshot.Capture
- Screenshot.IsInProgress
- Screenshot.Succeeded
- MediaLibrary.ExportImage

Screenshot.IsInProgress

Brief

Query whether a screen capture is still completing.

Definition

```
boolean Screenshot.IsInProgress(Screenshot shot)
```

Arguments

shot - the screenshot object to check

Return Values

true if still capturing, false if finished (either successfully or with an error)

Description

Query whether a screen capture initiated with Screenshot.Capture() has completed. If this returns false an error may also have occurred; check the value of Screenshot.Succeeded() to determine this. Once this returns false it is then safe to set the screenshot object to nil if it is no longer required.

Examples

```
if (shot:IsInProgress()) then
    print('Capturing...')
elseif (shot:Succeeded()) then
    -- do something with screenshot
end
```

See Also

Screenshot.Succeeded

Brief

Query whether a screen capture completed successfully.

Definition

```
boolean Screenshot.Succeeded(Screenshot shot)
```

Arguments

shot - the screenshot object to check

Return Values

true if capture completed successfully or false if still in progress or encountered an error

Description

Query whether a screen capture initiated with Screenshot.Capture() completed successfully. If the capture is still in progress false will be returned.

Examples

```
if (shot:Succeeded()) then
    MediaLibrary.ExportImage(shot, "Home Screenshot")
end
```

See Also

None

Sound

Sound.Create

Brief

Creates an empty sound handle.

Definition

```
Sound Sound.Create()
```

Arguments

None

Return Values

A fresh sound handle.

Description

The returned sound handle can be used repeatedly, removing the allocation cost associated with playing sounds from the SoundBank library.

Examples

```
mySound = Sound.Create()
myBank = SoundBank.Find("mySounds")
mySound:Play3d(myBank, "bang", LocalPlayer.GetPosition(), 1, 0, 5, 8, 0, 0, 0, 0)
```

See Also

- SoundBank.Play3d
- SoundBank.Play2d

Sound.GetVolume

Brief

Retrieves the volume of a sound.

Definition

```
number Sound.GetVolume( Sound sound )
```

Arguments

sound - The sound object.

Return Values

A floating point value between 0.0 (silent) and 1.0 (full volume).

Description

Retrieves the volume of a sound.

Examples

```
volume = Sound.GetVolume( mySound )
```

See Also

- Sound.SetVolume

Sound.IsPaused

Brief

Reports if the sound is paused or not.

Definition

```
boolean Sound.IsPaused( Sound sound )
```

Arguments

sound - The sound object.

Return Values

Returns true if the sound is paused and false otherwise.

Description

Returns true if the sound is paused and false otherwise.

Examples

```
if ( Sound.IsPaused( mySound ) ) then
    Sound.Resume( mySound )
end
```

See Also

- Sound.Pause
- Sound.Resume

Sound.IsPlaying

Brief

Reports if the sound is playing or not.

Definition

```
boolean Sound.IsPlaying( Sound sound )
```

Arguments

sound - The sound object.

Return Values

Returns true if the sound is playing and false otherwise.

Description

Returns true if the sound is playing and false otherwise.

Examples

```
if ( Sound.IsPlaying( mySound ) ) then
    Sound.Stop( mySound )
end
```

See Also

- Sound.Stop

Sound.IsValid

Brief

Can be used to check if a sound is valid.

Definition

```
boolean Sound.IsValid()
```

Arguments

None

Return Values

true if the sound is valid, false otherwise.

Description

Can be used to check if a sound is valid. A sound can become invalid if its SoundBank has been released.

Examples

```
if Sound.IsValid( mySound ) == false then
    mySound = nil
end
```

See Also

- Resource.Release

Sound.Pause

Brief

Pauses the sound.

Definition

```
Sound.Pause( Sound sound )
```

Arguments

sound - The sound object.

Return Values

None

Description

Pauses the sound.

Examples

```
if ( Sound.IsPlaying( mySound ) ) then
    Sound.Pause( mySound )
end
```

See Also

- Sound.IsPlaying

Sound.Play2d

Brief

Plays a named sound from a sound bank.

Definition

```
Sound.Play2d( Sound sound, SoundBank soundbank, string name, number volume = 1.0, number pan = 0.0,
number pitch = 0.0, number r0 = 0.0, number r1 = 0.0, number r2 = 0.0, number r3 = 0.0  )
```

Arguments

soundbank - The soundbank to play the sound fromname - The name of the sample within the SoundBank as set within SCREAM.volume - Value between 0.0 and 1.0 for controlling the volume of the sound.pan - Value between -1.0 and 1.0 for controlling the stereo output of the sound.pitch - This parameter will be implemented in the next release of the HDK.r0 - Passed to your scream sound's register 0.r1 - Passed to your scream sound's register 1.r2 - Passed to your scream sound's register 2.r3 - Passed to your scream sound's register 3.

Return Values

None

Description

Plays a named sound in the sound bank.

Examples

```
soundbank = SoundBank.Find( "game_bank" )
dingSound = Sound.Create()
dingSound:Play2d( soundbank, "ding" )
```

See Also

- Sound.Play3d
- Sound.Create
- Sound.Play2d

Sound.Play3d

Brief

Plays a named 3D sound from a sound bank.

Definition

```
Sound.Play3d( Sound sound, SoundBank soundbank, string name, Vector4 position, number volume = 1.0,
number pitch = 0.0, number falloffStart = 0.0, number falloffEnd = 1.0, number r1 = 0.0, number r2 =
0.0, number r3 = 0.0, number r4 = 0.0)
```

Arguments

soundbank - The soundbank to play the sound fromname - The name of the sample within the SoundBank as set within SCREAM.position - 3D world space coordinate of the sound (w-component is ignored).volume - Value between 0.0 and 1.0 for controlling the volume of the sound.pitch - This parameter will be implemented in the next release of the HDK.falloffStart - Value greater than 0.0 for controlling the distance from the position parameter at which the sound's volume begins to linearly attenuate.falloffEnd - Value greater than or equal to falloffStart for controlling the distance from the position parameter at which the sound's volume falls to zero.r0 - Passed to your scream sound's register 0.r1 - Passed to your scream sound's register 1.r2 - Passed to your scream sound's register 2.r3 - Passed to your scream sound's register 3.

Return Values

None

Description

Plays a named 3D sound from the sound bank.

Examples

```
soundbank = SoundBank.Find( "game_bank" )
ballRollingSound = Sound.Create()
ballRollingSound:Play3d( soundbank, "roll", LocalPlayer.GetPosition(), 1.0, 0.0, 10.0, 20.0)
```

See Also

- Sound.Play2d
- Sound.Create
- Sound.Play3d

Sound.Resume

Brief

Resumes a paused sound.

Definition

```
Sound.Resume( Sound sound )
```

Arguments

sound - The sound object.

Return Values

None

Description

Resumes a paused sound.

Examples

```
if ( Sound.IsPaused( mySound ) ) then
    Sound.Resume( mySound )
end
```

See Also

- Sound.IsPaused

Sound.SetPosition

Brief

Set the 3D world position of the sound

Definition

```
Sound.SetPosition( Sound sound, Vector4 position )
```

Arguments

sound - The sound object.position - 3D world space coordinate of the sound (w-component is ignored)

Return Values

None

Description

Set the 3D world position of the sound

Examples

```
Sound.SetPosition( sound, Vector4.Create(0, 10, 0, 0) )
```

See Also

- SoundBank.Play3d

Sound.SetRegister

Brief

Sets the specified register's value to the specified value.

Definition

```
Sound.SetRegister( Sound sound, number register, number value )
```

Arguments

sound - The sound object.register - Which register (0 - 3) to setvalue - the value to set the specified register to

Return Values

Description

Sets the specified register's value to the specified value.

Examples

```
Sound.SetRegister( sound, 2, 10 )
```

See Also

- SoundBank.Play2d
- SoundBank.Play3d

Sound.SetVolume

Brief

Sets the volume of a sound.

Definition

```
Sound.SetVolume( Sound sound, number volume )
```

Arguments

sound - The sound object.volume - floating point value between 0.0 (silent) and 1.0 (full volume)

Return Values

None

Description

Sets the volume of a sound retrieved from a sound bank.

Examples

```
Sound.SetVolume( mySound, 0.5 )
```

See Also

- Sound.GetVolume

Sound.Stop

Brief

Stops the sound playing.

Definition

```
Sound.Stop( Sound sound )
```

Arguments

sound - The sound object.

Return Values

None

Description

Stops the sound playing.

Examples

```
if ( Sound.IsPlaying( mySound ) ) then
    Sound.Stop( mySound )
end
```

See Also

- Sound.IsPlaying

SoundBank

SoundBank.Find

Brief

Looks up and returns the sound bank with the specified name.

Definition

```
SoundBank SoundBank.Find( string name, ResourcePack rp = nil )
```

Arguments

name - The string name associated with the sound bank as specified in Object Editor.rp - The ResourcePack to search.

Return Values

The sound bank instance associated with the specified string name.

Description

Looks up and returns the sound bank with the specified name. Returns nil if the sound bank is not found.

Examples

```
soundbank = SoundBank.Find( "game_bank" )
```

See Also

- SoundBank.Play2d
- SoundBank.Play3d

SoundBank.Play2d

Brief

Plays a named sound in the sound bank.

Definition

```
Sound SoundBank.Play2d( SoundBank soundbank, string name, number volume = 1.0, number pan = 0.0,
number pitch = 0.0, number r0 = 0.0, number r1 = 0.0, number r2 = 0.0, number r3 = 0.0  )
```

Arguments

name - The name of the sample within the SoundBank as set within SCREAM.volume - Value between 0.0 and 1.0 for controlling the volume of the sound.pan - Value between -1.0 and 1.0 for controlling the stereo output of the sound.pitch - Values different to 0 represent an octave multiplier, 1 is 1 octave higher, -1 is 1 octave lower, 2 is 2 octaves higher, and so on.r0 - Passed to your scream sound's register 0.r1 - Passed to your scream sound's register 1.r2 - Passed to your scream sound's register 2.r3 - Passed to your scream sound's register 3.

Return Values

Returns a Sound to handle the instance of the new sound.

Description

Plays a named sound in the sound bank.

Examples

```
soundbank = SoundBank.Find( "game_bank" )
dingSound = SoundBank.Play2d( soundbank, "ding" )
```

See Also

- SoundBank.Find
- SoundBank.Play3d

SoundBank.Play3d

Brief

Plays a named 3D sound in the sound bank.

Definition

```
Sound SoundBank.Play3d( SoundBank soundbank, string name, Vector4 position, number volume = 1.0,
number pitch = 0.0, number falloffStart = 0.0, number falloffEnd = 1.0, number r1 = 0.0, number r2 =
0.0, number r3 = 0.0, number r4 = 0.0  )
```

Arguments

name - The name of the sample within the SoundBank as set within SCREAM.position - 3D world space coordinate of the sound (w-component is ignored).volume - Value between 0.0 and 1.0 for controlling the volume of the sound.pitch - Values different to 0 represent an octave multiplier, 1 is 1 octave higher, -1 is 1 octave lower, 2 is 2 octaves higher, and so on.falloffStart - Value greater than 0.0 for controlling the distance from the position parameter at which the sound's volume begins to linearly attenuate.falloffEnd - Value greater than or equal to falloffStart for controlling the distance from the position parameter at which the sound's volume falls to zero.r0 - Passed to your scream sound's register 0.r1 - Passed to your scream sound's register 1.r2 - Passed to your scream sound's register 2.r3 - Passed to your scream sound's register 3.

Return Values

Returns a Sound to handle the instance of the new sound.

Description

Plays a named 3D sound in the sound bank.

Examples

```
soundbank = SoundBank.Find( "game_bank" )
bombSound = SoundBank.Play3d( soundbank, "explosion", Vector4.Create(0, 10, 0, 0), 1.0, 0.0, 10.0,
20.0)
```

See Also

- SoundBank.Find
- SoundBank.Play2d

SoundStream

SoundStream.Find

Brief

Finds a named SoundStream in the script's list of resources.

Definition

```
SoundStream SoundStream.Find( string name )
```

Arguments

name - the string name associated with the SoundStream as specified in an object's resources.xml file.

Return Values

The sound stream instance associated with the specified string name.

Description

Finds a named SoundStream in the script's list of resources.

Examples

```
bgmusic = SoundStream.Find( "music" )
```

See Also

SoundStream.GetCurrentTime

Brief

Get the current playback time for the stream.

Definition

```
number SoundStream.GetCurrentTime( SoundStream stream )
```

Arguments

stream - The sound stream object.

Return Values

time - The current playback time of the stream in seconds.

Description

Get the current playback time for the stream.

Examples

```
bgmusic = SoundStream.Find( "music" )
time = SoundStream.GetCurrentTime()
```

See Also

- SoundStream.SetCurrentTime

SoundStream.GetTotalTime

Brief

Get the total playback time for the stream.

Definition

```
number SoundStream.GetTotalTime( SoundStream stream )
```

Arguments

stream - The sound stream object.

Return Values

time - The total time of the stream in seconds.

Description

Get the total playback time for the stream.

Examples

```
bgmusic = SoundStream.Find( "music" )
time = SoundStream.GetTotalTime()
```

See Also

- SoundStream.GetCurrentTime

SoundStream.GetVolume

Brief

Get the sound stream's volume.

Definition

```
number SoundStream.GetVolume( SoundStream stream )
```

Arguments

stream - The sound stream object.

Return Values

volume - value between 0.0 and 1.0

Description

Get the sound stream's volume.

Examples

```
bgmusic = SoundStream.Find( "music" )
volume = SoundStream.GetVolume( bgmusic )
```

See Also

- SoundStream.SetVolume

SoundStream.IsPaused

Brief

Indicates whether the stream is paused

Definition

```
SoundStream.IsPaused( SoundStream stream )
```

Arguments

stream - The sound stream object.

Return Values

true if the stream is paused

Description

Indicates whether the stream is paused

Examples

```
idLoaded = SoundStream.IsPaused()
```

See Also

SoundStream.IsPlaying

Brief

Indicates whether the stream is playing

Definition

```
SoundStream.IsPlaying( SoundStream stream )
```

Arguments

stream - The sound stream object.

Return Values

true if the stream is playing

Description

Indicates whether the stream is playing

Examples

```
idLoaded = SoundStream.IsPlaying()
```

See Also

SoundStream.Pause

Brief

Pause the sound stream.

Definition

```
SoundStream.Pause( SoundStream stream )
```

Arguments

stream - The sound stream object.

Return Values

None

Description

Pause the sound stream.

Examples

```
bgmusic = SoundStream.Find( "music" )
SoundStream.Pause( bgmusic )
```

See Also

- SoundStream.Play2d
- SoundStream.Play3d

SoundStream.Play2d

Brief

Starts playing the sound stream.

Definition

```
SoundStream.Play2d( SoundStream stream, boolean looping = false, number volume = 1.0 )
```

Arguments

stream - The sound stream object.looping - true to loop the stream, false to play it once.volume - value between 0.0 and 1.0 for controlling the volume of the stream.

Return Values

None

Description

Starts playing the sound stream. Volume of 1.0 is 100% volume.

Examples

```
bgmusic = SoundStream.Find( "music" )
SoundStream.Play2d( bgmusic, true, 1.0 )
```

See Also

- SoundStream.Stop

SoundStream.Play3d

Brief

Starts playing the sound stream at a 3D position in the world

Definition

```
SoundStream.Play3d( SoundStream stream, boolean looping = false, number volume = 1.0, Vector4 position
= (0,0,0,0), number falloffStart = 0, number falloffEnd = 1 )
```

Arguments

stream - The sound stream object.looping - true to loop the stream, false to play it once.volume - value between 0.0 and 1.0 for controlling the volume of the stream.position - 3D world space coordinate of the sound (w-component is ignored)falloffStart - Value greater than or equal to 0.0 for controlling the distance from the position parameter at which the sound's volume begins to linearly attenuate.falloffEnd - Value greater than or equal to falloffStart for controlling the distance from the position parameter at which the sound's volume falls to zero.

Return Values

None

Description

Starts playing the sound stream at a 3D position in the world. Volume of 1.0 is 100% volume.

Examples

```
bgmusic = SoundStream.Find( "music" )
SoundStream.Play3d( bgmusic, true, 1.0, Vector4.Create(0, 10, 0, 0), 10, 20 )
```

See Also

- SoundStream.Stop

SoundStream.Resume

Brief

Resume the sound stream.

Definition

```
SoundStream.Resume( SoundStream stream )
```

Arguments

stream - The sound stream object.

Return Values

None

Description

Resume the sound stream.

Examples

```
bgmusic = SoundStream.Find( "music" )
SoundStream.Resume( bgmusic )
```

See Also

- SoundStream.Play2d
- SoundStream.Play3d

SoundStream.SetCurrentTime

Brief

Set the current playback time for the stream.

Definition

```
boolean SoundStream.SetCurrentTime( SoundStream stream, number time )
```

Arguments

stream - The sound stream object.time - the time (in seconds) to set

Return Values

result - true if setted false if not

Description

Set the current playback time for the stream.

Examples

```
bgmusic = SoundStream.Find( "music" )
result = SoundStream.SetCurrentTime( 10 )
```

See Also

- SoundStream.GetCurrentTime

SoundStream.SetPosition

Brief

Set the 3D world position of the stream

Definition

```
SoundStream.SetPosition( SoundStream stream, Vector4 position )
```

Arguments

stream - The sound stream object.position - 3D world space coordinate of the sound (w-component is ignored)

Return Values

None

Description

Set the 3D world position of the stream

Examples

```
bgmusic = SoundStream.Find( "music" )
SoundStream.SetPosition( bgmusic, Vector4.Create(0, 10, 0, 0) )
```

See Also

- SoundStream.Play3d

SoundStream.SetVolume

Brief

Set the sound stream's volume.

Definition

```
SoundStream.SetVolume( SoundStream stream, number volume )
```

Arguments

stream - The sound stream object.volume - value between 0.0 and 1.0 for controlling the volume of the stream.

Return Values

None

Description

Set the sound stream's volume.

Examples

```
bgmusic = SoundStream.Find( "music" )
SoundStream.SetVolume( bgmusic, 0.5 )
```

See Also

- SoundStream.GetVolume

SoundStream.Stop

Brief

Stops the sound stream.

Definition

```
SoundStream.Stop( SoundStream stream )
```

Arguments

stream - The sound stream object.

Return Values

None

Description

Stops the sound stream.

Examples

```
bgmusic = SoundStream.Find( "music" )
SoundStream.Stop( bgmusic )
```

See Also

- SoundStream.Play2d
- SoundStream.Play3d

Sprite

Sprite.Create

Brief

Creates a new Sprite object.

Definition

```
Sprite Sprite.Create()
```

Arguments

None

Return Values

A new Sprite instance.

Description

Creates a new Sprite object.

Examples

```
sprite = Sprite.Create()
```

See Also

- Renderer.DrawSprite

Sprite.GetColor

Brief

Returns the color of the sprite as a Vector4.

Definition

```
Vector4 Sprite.GetColor( Sprite sprite )
Sprite.GetPosition( Sprite sprite, Vector4 outColor )
```

Arguments

sprite - The sprite to query.outColor - (out) The vector to receive the color.

Return Values

A Vector4 containing the RGBA values (in that order) of the sprite.

Description

Returns the color of the sprite as a Vector4 where x is red, y, is green, z is blue, w is alpha. They range from 0 to 1 where 1 is the maximum color.

Examples

```
col = Sprite.GetColor( sprite )
print( col )
```

See Also

- Sprite.SetColor

Sprite.GetElapsedAnimTime

Brief

Returns the elapsed animation time in seconds.

Definition

```
number Sprite.GetElapsedAnimTime( Sprite sprite )
```

Arguments

sprite - The sprite to query.

Return Values

The elapsed time in seconds of the sprite's animation.

Description

Returns the elapsed animation time in seconds.

Examples

```
print( Sprite.GetElapsedAnimTime( sprite ) )
```

See Also

- Sprite.GetTotalAnimTime
- Sprite.IsAnimPaused
- Sprite.PlayAnim

- Sprite.ResumeAnim
- Sprite.StopAnim

Sprite.GetPosition

Brief

Returns the screen space coordinate of a sprite.

Definition

```
Vector4 Sprite.GetPosition( Sprite sprite )
Sprite.GetPosition( Sprite sprite, Vector4 outPos )
```

Arguments

sprite - The sprite to query.outPos - (out) The vector to receive the position.

Return Values

A Vector4 object containing the screen space position of the sprite. Only the X and Y members will be valid.

Description

Returns the position of the sprite. Note that this vector is a copy created by the function and so changing its value will not change the sprite.

Examples

```
pos = Sprite.GetPosition( sprite )
print( pos )
```

See Also

- Sprite.SetPosition

Sprite.GetRotation

Brief

Returns the angle of rotation of the sprite.

Definition

```
number Sprite.GetRotation( Sprite sprite )
```

Arguments

sprite - The sprite to query.

Return Values

A floating point value containing the rotation in degrees of the sprite.

Description

Returns the angle of rotation in degrees of the sprite.

Examples

```
rot = Sprite.GetRotation( sprite )
print( rot .. " degrees" )
```

See Also

- Sprite.SetRotation

Sprite.GetScale

Brief

Returns as a Vector4 the scale of the sprite.

Definition

```
Vector4 Sprite.GetScale( Sprite sprite )
Sprite.GetScale( Sprite sprite, Vector4 outScale )
```

Arguments

sprite - The sprite to query.outScale - (out) The vector to receive the scale.

Return Values

A Vector4 instance containing the sprite's scale. Only the X and Y members will be valid.

Description

Returns the scale of the sprite. Note that this vector is a copy of the vector and so changing its value will not change the sprite.

Examples

```
scale = Sprite.GetScale( sprite )
print( scale )
```

See Also

- Sprite.GetScale

Sprite.GetTotalAnimTime

Brief

Returns the total animation time in seconds.

Definition

```
number Sprite.GetTotalAnimTime( Sprite sprite )
```

Arguments

sprite - The sprite to query.

Return Values

The length in seconds of the sprite's animation.

Description

Returns the total animation time in seconds. If the animation is looping it returns the total time for one loop of the animation.

Examples

```
print( Sprite.GetTotalAnimTime( sprite ) )
```

See Also

- Sprite.GetElapsedAnimTime
- Sprite.IsAnimPaused
- Sprite.PlayAnim
- Sprite.ResumeAnim
- Sprite.StopAnim

Sprite.IsAnimPaused

Brief

Reports if the animation is paused.

Definition

```
boolean Sprite.IsAnimPaused( Sprite sprite )
```

Arguments

sprite - The sprite to query.

Return Values

true if the sprite's animation is paused, false otherwise.

Description

Returns true if the sprite's animation is paused (using Sprite.PauseAnim) and false otherwise.

Examples

```
if Sprite.IsAnimPaused( sprite ) then
    print( "Animation is paused" )
end
```

See Also

- Sprite.GetElapsedAnimTime
- Sprite.GetTotalAnimTime
- Sprite.PlayAnim
- Sprite.ResumeAnim
- Sprite.StopAnim

Sprite.IsAnimPlaying

Brief

Reports if an animation is playing on the sprite.

Definition

```
boolean Sprite.IsAnimPlaying( Sprite sprite )
```

Arguments

sprite - The sprite to query.

Return Values

true if an animation is playing and false otherwise.

Description

Returns true if an animation is playing and false otherwise. Note that it is still considered as playing even if the animation is paused.

Examples

```
if Sprite.IsAnimPlaying( sprite ) then
    print( "Playing an animation" )
end
```

See Also

- Sprite.GetElapsedAnimTime
- Sprite.GetTotalAnimTime
- Sprite.PlayAnim
- Sprite.ResumeAnim
- Sprite.StopAnim

Sprite.PauseAnim

Brief

Pauses an animation.

Definition

```
Sprite.PauseAnim( Sprite sprite )
```

Arguments

sprite - The sprite to update.

Return Values

None.

Description

Pauses an animation. It can then be resumed with ResumeAnim.

Examples

```
Sprite.PauseAnim( sprite )
```

See Also

- Sprite.GetElapsedAnimTime
- Sprite.GetTotalAnimTime
- Sprite.PlayAnim
- Sprite.ResumeAnim
- Sprite.StopAnim

Sprite.PlayAnim

Brief

Plays a texture animation on the sprite.

Definition

```
Sprite.PlayAnim( Sprite sprite, SpriteAnim anim, number framesPerSecond = 1.0 )
```

Arguments

sprite - The sprite to update.anim - The animation to apply on the sprite.framesPerSecond - The number of animation frames to be displayed every second.

Return Values

None.

Description

Plays a texture animation with the specified anim at the given animation speed. The Sprites UVs are overwritten with the UVs from the animation.

Examples

```
Sprite.PlayAnim( sprite, anim, 30 )
```

See Also

- Sprite.GetElapsedAnimTime
- Sprite.GetTotalAnimTime
- Sprite.PauseAnim
- Sprite.ResumeAnim
- Sprite.StopAnim

Sprite.ResumeAnim

Brief

Resumes a paused animation.

Definition

```
Sprite.ResumeAnim( Sprite sprite )
```

Arguments

sprite - The sprite to update.

Return Values

None.

Description

Resumes a paused animation.

Examples

```
Sprite.ResumeAnim( sprite )
```

See Also

- Sprite.GetElapsedAnimTime
- Sprite.GetTotalAnimTime
- Sprite.PauseAnim
- Sprite.PlayAnim
- Sprite.StopAnim

Sprite.SetColor

Brief

Sets the color of the sprite.

Definition

```
Sprite.SetColor( Sprite sprite, Vector4 rgba )
Sprite.SetColor( Sprite sprite, number red = 1, number green = 1, number blue = 1, number alpha = 1 )
```

Arguments

sprite - The sprite to update.rgba - A Vector4 containing the RGBA values where each component must be between 0.0 and 1.0.red - The red component, ranges from 0 to 1. Defaults to 1.green - The green component, ranges from 0 to 1. Defaults to 1.blue - The blue component, ranges from 0 to 1. Defaults to 1.alpha - The alpha component, ranges from 0 to 1. Defaults to 1.

Return Values

None.

Description

Sets the color of the sprite.

Examples

```
redHalfAlpha = Vector4.Create( 1, 0, 0, 0.5 )
sprite = Sprite.Create()
Sprite.SetColor( sprite, redHalfAlpha )
```

See Also

- Sprite.GetColor

Sprite.SetPosition

Brief

Sets the position of the sprite.

Definition

```
Sprite.SetPosition( Sprite sprite, Vector4 position )
Sprite.SetPosition( Sprite sprite, number x = 0, number y = 0 )
```

Arguments

sprite - The sprite to update.position - The new sprite position, where only the X and Y members are used.x - The X component of the sprite position.y - The Y component of the sprite position.

Return Values

None.

Description

Sets the screen coordinate of the sprite. The position specifies the coordinate of the center of the sprite.

Examples

```
pos = Vector4.Create( 10, 10 )
sprite = Sprite.Create()
Sprite.SetPosition( sprite, pos )
```

See Also

- Sprite.GetPosition

Sprite.SetRotation

Brief

Sets the rotation of the sprite.

Definition

```
Sprite.SetRotation( Sprite sprite, number rotation )
```

Arguments

sprite - The sprite to update.rotation - The new sprite rotation (in degrees).

Return Values

None.

Description

Sets the rotation of the sprite in degrees.

Examples

```
sprite = Sprite.Create()
Sprite.SetRotation( sprite, 90.0 )
```

See Also

- Sprite.GetRotation

Sprite.SetScale

Brief

Sets the scale of the sprite.

Definition

```
Sprite.SetScale( Sprite sprite, Vector4 vector )
Sprite.SetScale( Sprite sprite, number x = 1, number y = 1 )
```

Arguments

sprite - The sprite to update.vector - The new sprite scale, where only the X and Y members are used.x - The horizontal sprite scale.y - The vertical sprite scale.

Return Values

None.

Description

Sets the x and y scale of the sprite.

Examples

```
scale = Vector4.Create( 10, 10 )
sprite = Sprite.Create()
Sprite.SetScale( sprite, scale )
```

See Also

- Sprite.GetScale

Sprite.SetTexture

Brief

Sets the current texture of the sprite.

Definition

```
Sprite.SetTexture( Sprite sprite, Texture texture )
```

Arguments

sprite - The sprite to update.texture - An instance of the Texture class that will represent the graphic of the sprite.

Return Values

None.

Description

Sets the current texture of the sprite. Note that this texture is replaced if an animation is played on the sprite. The UVs are set to default, i.e. the whole of the texture.

Examples

```
sprite = Sprite.Create()
Sprite.SetTexture( sprite, Texture.Find( "SpriteTexture" ) )
```

See Also

- Sprite.SetUvs

Sprite.SetUvs

Brief

Sets the UVs of the sprite.

Definition

```
Sprite.SetUvs( Sprite sprite, number tlu, number tlv, number bru, number brv )
Sprite.SetUvs( Sprite sprite, Vector4 tlbr )
Sprite.SetUvs( Sprite sprite, number tlu, number tlv, number blu, number blv, number bru, number brv,
number tru, number trv )
Sprite.SetUvs( Sprite sprite, Vector4 tlbl, Vector4 brtr )
```

Arguments

sprite - The sprite to update.tlu - The u-component of the top left texture coordinate for the sprite.tlv - The v-component of the top left texture coordinate for the sprite.blu - The u-component of the bottom left texture coordinate for the sprite.blv - The v-component of the bottom left texture coordinate for the sprite.bru - The u-component of the bottom right texture coordinate for the sprite.brv - The v-component of the bottom right texture coordinate for the sprite.tru - The u-component of the top right texture coordinate for the sprite.trv - The v-component of the top right texture coordinate for the sprite.tlbr - A vector containing the top left and bottom right UV coordinates for the sprite.tlbl - A vector containing the top left and bottom left UV coordinates for the sprite.brtr - A vector containing the bottom right and top right UV coordinates for the sprite.

Return Values

None.

Description

Sets the UVs of the sprite. If passing in 4 numbers either as numbers or packed inside a Vector4 it assumes a rectangle and generates the two other uvs from the two uvs specified. If 4 uvs are supplied it uses them as is. The default uvs for a Sprite are [0,0],[0,1],[1,1],[1,0]. Also, UVs are set to the default when Sprite.SetTexture is called or Sprite.PlayAnim.

Examples

```
sprite = Sprite.Create()
Sprite.SetTexture( sprite, texture )

-- Example 1: Set to default UVs using each prototype
Sprite.SetUvs( sprite, 0, 0, 1, 1 )
Sprite.SetUvs( sprite, Vector4.Create( 0, 0, 1, 1 ) )
Sprite.SetUvs( sprite, 0, 0, 0, 1, 1, 1, 1, 0 )
Sprite.SetUvs( sprite, Vector4.Create( 0, 0, 0, 1 ), Vector4.Create( 1, 1, 1, 0 ) )

-- Example 2: Set the sprite to use the top left quadrant of the texture
Sprite.SetUvs( sprite, 0, 0, 0.5, 0.5 )
```

See Also

- Sprite.SetTexture
- Sprite.PlayAnim

Sprite.StopAnim

Brief

Stops the current animation if it is playing.

Definition

```
Sprite.StopAnim( Sprite sprite )
```

Arguments

sprite - The sprite to update.

Return Values

None.

Description

Stops the current animation if it is playing. The sprite's texture is left at whatever was last displayed.

Examples

```
Sprite.StopAnim( sprite )
```

See Also

- Sprite.GetElapsedAnimTime
- Sprite.GetTotalAnimTime
- Sprite.PauseAnim
- Sprite.PlayAnim
- Sprite.ResumeAnim

SpriteAnim

SpriteAnim.AddFrames

Brief

Adds one or more new frames to an animation.

Definition

```
SpriteAnim.AddFrames( SpriteAnim anim, Texture texture, number frameWidth = textureWidth, number
frameHeight = textureHeight )
```

Arguments

texture - the new frame of animation.frameWidth - the width of the frame (defaults to the width of the specified texture).frameHeight - the width of the frame (defaults to the height of the specified texture).

Return Values

None.

Description

Adds one or more new frames to an animation. If frameWidth and/or frameHeight is specified it subdivides the texture up into several frames, where frames are in left to right and then top to bottom order.

Examples

```
-- Create a 2 frame sprite animation
anim = SpriteAnim.Create()
SpriteAnim.AddFrames( anim, Texture.Find( "image1" ) )
SpriteAnim.AddFrames( anim, Texture.Find( "image2" ) )

-- Loop the animation every second
sprite = Sprite.Create()
Sprite.PlayAnim( sprite, anim, 2 )
```

See Also

- Sprite.Create
- Sprite.PlayAnim
- SpriteAnim.Create

SpriteAnim.Create

Brief

Creates a new SpriteAnim object.

Definition

```
SpriteAnim SpriteAnim.Create()
```

Arguments

None

Return Values

A new SpriteAnim instance.

Description

Creates a new SpriteAnim object.

Examples

```
anim = SpriteAnim.Create()
```

See Also

- SpriteAnim.AddFrames

SpriteAnim.IsLooping

Brief

Reports if the animation is looping or not.

Definition

```
boolean SpriteAnim.IsLooping( SpriteAnim anim )
```

Arguments

anim - The srite anim object.

Return Values

Returns true if the animation has been set to looping and false otherwise.

Description

Reports if the animation is looping or not.

Examples

```
print( SpriteAnim.IsLooping( myAnim ) )
```

See Also

- SpriteAnim.SetLooping

SpriteAnim.SetLooping

Brief

Sets the animation looping status.

Definition

```
SpriteAnim.SetLooping( SpriteAnim anim, boolean value )
```

Arguments

value - true to make the animation to loop, false to make the animation play once.

Return Values

None.

Description

Sets the animation to loop or not.

Examples

```
anim = SpriteAnim.Create()
SpriteAnim.SetLooping( anim, true )
```

See Also

- SpriteAnim.IsLooping

StreamOps

StreamOps.AddFloat32

Brief

Add two float buffers or add a scalar onto a float buffer.

Definition

```
StreamOps.AddFloat32(MemoryContainer dest, number destOffset, MemoryContainer src1, number src1Offset,
                     MemoryContainer src2, number src2Offset, number count, number stride = 1)
StreamOps.AddFloat32(MemoryContainer dest, number destOffset, MemoryContainer src1, number src1Offset,
                     number scalar, number count, number stride = 1)
```

Arguments

dest - Destination memory container.destOffset - Offset into destination memory container at which to start writing results.src1 - First source memory container.src1Offset - Offset into first source memory container at which to start reading source data.src2 - Second source memory container.src2Offset - Offset into second source memory container at which to start reading source data.count - Number of floats to process.stride - Number of floats to increment offset by. Used for both sources and destination.scalar - A scalar to add to all source elements.

Return Values

None.

Description

Note that all offsets and strides are in "elements", float32s in this case. This function is equivalent to:
local offset = 0
for i = 1, count do
dest:SetFloat32(destOffset * 4 + offset, src1:GetFloat32(src1Offset * 4 + offset) + src2:GetFloat32(src2Offset * 4 + offset))
– OR dest:SetFloat32(destOffset * 4 + offset, src1:GetFloat32(src1Offset * 4 + offset) + scalar)
offset = offset + stride * 4
end

Examples

```

```

See Also

- StreamOps.SubFloat32
- StreamOps.MulFloat32
- StreamOps.DivFloat32

StreamOps.BoundsVec4

Brief

Compute the bounds of a list of Vector4 values.

Definition

```
StreamOps.BoundsVec4(MemoryContainer src, number srcOffset, number count, Vector4 min, Vector4 max,
number stride = 1)
```

Arguments

src - Source memory container. Operation will be faster if 16-byte aligned.srcOffset - Offset into source memory container at which to start processing.count - Number of Vector4 values to process.min - Vector4 to return the minimum bound into. Assumed to be pre-set to the desired initial value.max - Vector4 to return the maximum bound into. Assumed to be pre-set to the desired initial value.stride - Number of Vector4s to increment offset by with each vector processed.

Return Values

None.

Description

Note that all offsets and strides are in "elements", Vector4s in this case. This function is equivalent to:
local offset = srcOffset * 4
local min = Vector4.Create()
local max = Vector4.Create()
local v = Vector4.Create()
for i = 1, count do
src:GetVector4(offset, v)
Vector4.Min(min, min, v)
Vector4.Max(max, max, v)
offset = offset + stride * 16
end
Note that the min and max parameters are not initialized to any value in this function, they should be initialized by the script to appropriate starting values for the bounds calculation.

Examples

```
local min = Vector4.Create(Const.MaxNum, Const.MaxNum, Const.MaxNum)
local max = Vector4.Create(-Const.MaxNum, -Const.MaxNum, -Const.MaxNum)
StreamOps.BoundsVec4(verts, 0, numVerts, min, max)
```

See Also

None

StreamOps.DivFloat32

Brief

Divide two float buffers or divide a float buffer by a scalar.

Definition

```
StreamOps.DivFloat32(MemoryContainer dest, number destOffset, MemoryContainer src1, number src1Offset,
                     MemoryContainer src2, number src2Offset, number count, number stride = 1)
StreamOps.DivFloat32(MemoryContainer dest, number destOffset, MemoryContainer src1, number src1Offset,
                     number scalar, number count, number stride = 1)
StreamOps.DivFloat32(MemoryContainer dest, number destOffset, number scalar, MemoryContainer src1,
number src1Offset,
                     number count, number stride = 1)
```

Arguments

dest - Destination memory container.destOffset - Offset into destination memory container at which to start writing results.src1 - First source memory container.src1Offset - Offset into first source memory container at which to start reading source data.src2 - Second source memory container.src2Offset - Offset into second source memory container at which to start reading source data.count - Number of floats to process.stride - Number of floats to increment offset by. Used for both sources and destination.scalar - A scalar to apply to each source element.

Return Values

None.

Description

Note that all offsets and strides are in "elements", float32s in this case. This function is equivalent to:
local offset = 0
for i = 1, count do
dest:SetFloat32(destOffset * 4 + offset, src1:GetFloat32(src1Offset * 4 + offset) / src2:GetFloat32(src2Offset * 4 + offset))
– OR dest:SetFloat32(destOffset * 4 + offset, src1:GetFloat32(src1Offset * 4 + offset) + scalar)
offset = offset + stride * 4
end

Examples

See Also

- StreamOps.SubFloat32
- StreamOps.AddFloat32
- StreamOps.MulFloat32

StreamOps.Float32ToUInt8

Brief

Convert 32-bit floating point values to unsigned 8-bit integer values.

Definition

```
StreamOps.Float32ToUInt8(MemoryContainer dest, number destOffset, MemoryContainer src, number
srcOffset, number count, number stride = 1, number multiplier = 1)
```

Arguments

dest - Destination memory container.destOffset - Offset into destination memory container at which to start storing results.src - Source memory container.srcOffset - Offset into source memory container at which to start reading values.count - Number of floats to process.stride - Number of floats to increment offset by. Used for source only.multiplier - Number to multiply float by prior to conversion.

Return Values

None.

Description

A common use for this function might be to convert float colors in the range [0, 1] to byte colors in the range [0, 255]. Note that all offsets and strides are in "elements", float32s in this case. This function is equivalent to:
local offset = srcOffset * 4
for i = 1, count do
dest:SetUInt8(destOffset, src:GetFloat32(offset) * multiplier)
offset = offset + stride * 4
destOffset = destOffset + 1 – int8
end

Examples

See Also

None

StreamOps.HorizAddFloat32

Brief

Sum values within a memory container.

Definition

```
number StreamOps.HorizAddFloat32(MemoryContainer src, number srcOffset, number count, number stride =
1)
```

Arguments

src - Source memory container.srcOffset - Offset into source memory container at which to start adding values.count - Number of floats to process.stride - Number of floats to increment offset by. Used for both sources and destination.

Return Values

The sum of the values.

Description

Note that all offsets and strides are in "elements", float32s in this case. This function is equivalent to
local offset = srcOffset * 4
local sum = 0
for i = 1, count do
sum = sum + src:GetFloat32(offset)
offset = offset + stride * 4
end

Examples

See Also

- StreamOps.AddFloat32

StreamOps.MulFloat32

Brief

Multiply two float buffers or multiply a float buffer by a scalar.

Definition

```
StreamOps.MulFloat32(MemoryContainer dest, number destOffset, MemoryContainer src1, number src1Offset,
                     MemoryContainer src2, number src2Offset, number count, number stride = 1)
StreamOps.MulFloat32(MemoryContainer dest, number destOffset, MemoryContainer src1, number src1Offset,
                     number scalar, number count, number stride = 1)
```

Arguments

dest - Destination memory container.destOffset - Offset into destination memory container at which to start writing results.src1 - First source memory container.src1Offset - Offset into first source memory container at which to start reading source data.src2 - Second source memory container.src2Offset - Offset into second source memory container at which to start reading source data.count - Number of floats to process.stride - Number of floats to increment offset by. Used for both sources and destination.scalar - A scalar to multiply each source element by.

Return Values

None.

Description

Note that all offsets and strides are in "elements", float32s in this case. This function is equivalent to:
local offset = 0
for i = 1, count do
dest:SetFloat32(destOffset * 4 + offset, src1:GetFloat32(src1Offset * 4 + offset) * src2:GetFloat32(src2Offset * 4 + offset))
– OR dest:SetFloat32(destOffset * 4 + offset, src1:GetFloat32(src1Offset * 4 + offset) + scalar)
offset = offset + stride * 4
end

Examples

See Also

- StreamOps.SubFloat32
- StreamOps.AddFloat32
- StreamOps.DivFloat32

StreamOps.ShufFloat32

Brief

Shuffle a float buffer based on a supplied index table.

Definition

```
StreamOps.ShufFloat32(MemoryContainer dest, number destOffset, MemoryContainer src1,
                      MemoryContainer indices, number count, number stride = 1)
```

Arguments

dest - Destination memory container.destOffset - Offset into destination memory container at which to start writing results.src1 - Source memory container.indices - Index memory container. Indices are float offsets into src1.count - Number of floats to process.stride - Number of floats to increment offset by. Used for both sources and destination.

Return Values

None.

Description

If the object has the debug flag set, all indices will be range-checked. Otherwise, it is the responsibility of the developer to ensure that all indices are valid and within the size of the source memory container. Note that all offsets and strides are in "elements", float32s in this case. This function is equivalent to:
local offset = 0
local indexOffset = 0
for i = 1, count do
local index = indices:GetInt16(indexOffset)
dest:SetFloat32(destOffset * 4 + offset, src1:GetFloat32(index * 4))
offset = offset + stride * 4
indexOffset = indexOffset + 2 – int16
end

Examples

See Also

None

StreamOps.ShufInt8

Brief

Shuffle an int8 buffer based on a supplied index table.

Definition

```
StreamOps.ShufInt8(MemoryContainer dest, number destOffset, MemoryContainer src1,
                       MemoryContainer indices, number count, number stride = 1)
```

Arguments

dest - Destination memory container.destOffset - Offset into destination memory container at which to start writing results.src1 - Source memory container.indices - Index memory container. Indices are 16-bit (int16) byte offsets into src1.count - Number of bytes to process.stride - Number of bytes to increment offset by. Used for both sources and destination.

Return Values

None.

Description

If the object has the debug flag set, all indices will be range-checked. Otherwise, it is the responsibility of the developer to ensure that all indices are valid and within the size of the source memory container. Note that all offsets and strides are in bytes (int8s). This function is equivalent to:
local offset = 0
local indexOffset = 0
for i = 1, count do
local index = indices:GetInt16(indexOffset)
dest:SetInt8(destOffset + offset, src1:GetInt8(index))
offset = offset + stride
indexOffset = indexOffset + 2 – int16
end

Examples

See Also

None

StreamOps.SubFloat32

Brief

Calculate difference of two float buffers or subtract a scalar from a float buffer.

Definition

```
StreamOps.SubFloat32(MemoryContainer dest, number destOffset, MemoryContainer src1, number src1Offset,
                     MemoryContainer src2, number src2Offset, number count, number stride = 1)
StreamOps.SubFloat32(MemoryContainer dest, number destOffset, MemoryContainer src1, number src1Offset,
                     number scalar, number count, number stride = 1)
StreamOps.SubFloat32(MemoryContainer dest, number destOffset, number scalar, MemoryContainer src1,
number src1Offset,
                     number count, number stride = 1)
```

Arguments

dest - Destination memory container.destOffset - Offset into destination memory container at which to start writing results.src1 - First source memory container.src1Offset - Offset into first source memory container at which to start reading source data.src2 - Second source memory container.src2Offset - Offset into second source memory container at which to start reading source data.count - Number of floats to process.stride - Number of floats to increment offset by. Used for both sources and destination.scalar - A scalar to apply to each source element.

Return Values

None.

Description

Note that all offsets and strides are in "elements", float32s in this case. This function is equivalent to:
local offset = 0
for i = 1, count do
dest:SetFloat32(destOffset * 4 + offset, src1:GetFloat32(src1Offset * 4 + offset) - src2:GetFloat32(src2Offset * 4 + offset))
– OR dest:SetFloat32(destOffset * 4 + offset, src1:GetFloat32(src1Offset * 4 + offset) + scalar)
offset = offset + stride * 4
end

Examples

```

```

See Also

- StreamOps.AddFloat32
- StreamOps.MulFloat32
- StreamOps.DivFloat32

System

System.AddChatLogMessage

Brief

Adds a lua message in the chat log.

Definition

```
System.AddChatLogMessage( string message, string nameOfSpeaker )
```

Arguments

message - The message to add to the chat log.nameOfSpeaker - The name of the speaker (optional).

Return Values

None.

Description

Adds a lua message in the chat log. If the chat log is not visible when this function is called, it will activate and display the message. The message text will appear in a different color to the other messages (r = 0.4, g = 0.7, b = 1.0).

Examples

```
me = LocalPlayer.GetPerson()
System.AddChatLogMessage( 'Good morning ' .. Person.GetName( me ) )
```

See Also

None

System.CommercePointIsClosed

Brief

Returns true if the commerce point is closed and false if it is open (active).

Definition

```
boolean System.CommercePointIsClosed()
```

Arguments

None

Return Values

Returns true if the commerce point is closed and false if it is open (active).

Description

Returns true if the commerce point is closed and false if it is open (active).

Examples

```
if System.CommercePointIsClosed() == true then
    System.CommercePointOpen( myUrl )
end
```

See Also

- System.CommercePointOpen

System.CommercePointOpen

Brief

Opens a commerce point with the specified url.

Definition

```
number System.CommercePointOpen( string url, string rootNodeId = nil, string initNodeId = nil )
```

Arguments

url - The url of the commerce point.rootNodeId - The "node id" of the commerce point xml element to treat as the "root node" of the commerce point.initNodeId - The "node id" of the commerce point xml element to treat as the "init node" of the commerce point.

Return Values

The result of the open request. Return values are:
1 - commerce point was opened
-1 - commerce point is already running

Description

Opens a commerce point with the specified url. See the return result to see if it failed or succeeded. The URL for the commerce point requires the correct path - these are set up using a standard format as follows:
SCENE_NAME/REGION_CODE/CP<REGION_CODE><CP_ShortNAME><2_CHAR_NUM>.xml

where:

SCENE_NAME is the name of the scene including the code on the end
e.g.
E_COMPA_DW1_5C2G_J5V
XFORM_Lounge_2_9KEL_1N8L
SCEE_PureFire_Lobby_0R2J_766

REGION_CODE is the PSN/PMT region code (SCEE=E, SCEA=A, SCEJ=J, SCEAsia=H)
CP_ShortNAME is a 'shortcode' that should be meaningful to you.
2_CHAR_NUM is simply a two character number which can mean anything, and again should be meaningful you (e.g. a version number)

When opening a commerce point on demand using the call, simply build the commerce point file path using the known information.

For example:

SCEA user opening a specific commerce point:
XFORM_Lounge_2_9KEL_1N8L/A/CPAXFL00.xml

where the CP_ShortNAME="XFL" and 2_CHAR_NUM="00"

SCEE user opening the same commerce point for their region:
XFORM_Lounge_2_9KEL_1N8L/E/CPEXFL00.xml If the rootNodeId parameter is specified, the "root node" will be set. If the root node is a <node> element, then navigation of the commerce point will be restricted to the browser page corresponding to that element, all child pages, the item info page and the basket. If the root node is a <object> or <product> element, then navigation of the commerce point will be restricted to the item info page and basket. If the initNodeId parameter is specified, the "init node" will be set. If the init node is a <node> element, then navigation of the commerce point will begin at the browser page corresponding to that element. If the init node is a <object> or <product> element, then navigation of the commerce point will begin at the item info page.

Examples

```
System.CommercePointOpen( 'XFORM_Lounge_2_9KEL_1N8L/E/CPEXFL00.xml' )
```

```
System.CommercePointOpen( 'XFORM_Lounge_2_9KEL_1N8L/E/CPEXFL00.xml', 'male headgear' )
```

```
System.CommercePointOpen( 'XFORM_Lounge_2_9KEL_1N8L/E/CPEXFL00.xml', 'male headgear', 'big hat' )
```

```
System.CommercePointOpen( 'XFORM_Lounge_2_9KEL_1N8L/E/CPEXFL00.xml', nil, 'big hat' )
```

See Also

- System.CommercePointIsClosed

System.DrawText2dInScene

Brief

Allow text to be displayed in 3D space.

Definition

```
System.DrawText2dInScene(string text, Vector4 position, Vector4 color,
                  Text3dAlign alignment, number size,
                  boolean isShadowed, Vector4 shadowColor,
                  boolean isBold, boolean isAlwaysOnTop, boolean isFixedSize)
```

Arguments

text - the text to be displayed.position - the 3D position at which the text will appear.color - the color of the text to be displayed.alignment - the alignment of the text, in relation to the position given.
This will be one of the following:

- Text3dAlignment.TopLeft
- Text3dAlignment.TopMiddle
- Text3dAlignment.TopRight
- Text3dAlignment.Left
- Text3dAlignment.Center
- Text3dAlignment.Right
- Text3dAlignment.BottomLeft
- Text3dAlignment.BottomMiddle
- Text3dAlignment.BottomRight
  Default is 'Text3dAlignCenter'.size - the size of the text. Default is '1.0',isShadowed - whether the text should have a drop shadow. Default is 'false'.shadowColor - the colour of the shadow, if 'isShadowed' is set to true. Default is black,
  if no color is provided.isBold - whether the text is in bold font. Default is 'false'.isAlwaysOnTop - whether the text should adhere to 3D geometry obscuring it. Default is 'false'.isFixedSize - whether the text should resize depending on distance away from camera. Default is 'false'.

Return Values

Description

Displays 2D text in 3D space, much like the avatar label user name.

Examples

```
System.DrawText2dInScene(text, position, color)
```

See Also

None

System.EnableIdleDisconnect

Brief

Sets whether the idle disconnect is allowed.

Definition

```
System.EnableIdleDisconnect(boolean enable)
```

Arguments

enable - True to enable or false to disable.

Return Values

None.

Description

If set to false, the user will not be disconnected if idle for too long. This can be useful for disabling the auto-disconnect while the user is watching a full-length video for example. The idle disconnect defaults to enabled.

Examples

```
System.EnableIdleDisconnect(false)
```

See Also

None

System.ExecTitle

Brief

Launches an installed title.

Definition

```
number System.ExecTitle( string titleId, number param, string errorCallback )
```

Arguments

titleId - The title ID of the title to launch.param - Integer value that will be passed through to the launched title.errorCallback - Callback function which will be invoked if the title fails to execute.

Return Values

0 if title not available, 1 otherwise.

Description

Launches an installed title. The integer value that is passed through to the game can be queried using the execData parameter of cellGameGetBootGameInfo. In the case where user confirmation is required before proceeding with ExecTitle, for example confirmation to leave a Game Launching session, 1 will be returned to indicate no errors so far and ExecTitle will only proceed if user confirms. If user cancels then the error callback will be invoked and this particular case should be handled. If the title fails to execute the callback will be invoked, within this callback use System.GetExecTitleErrorCode() to retreive the error code. Scripts that make use of this function must also include a game_launch_properties component in the object to allow PS Home to manage any potential system conflicts. Failure to comply will result in a Lua error being generated.

Examples

```
gameOptions = 10
if GameInfo.IsGamesListBusy() == false then
    System.ExecTitle( "XXXX12345", gameOptions, "MyLaunchErrorCB()" )
end
```

See Also

- GameInfo.IsGamesListBusy
- GameInfo.SearchByTitleId

- GameInfo.IsSearchFinished
- GameInfo.GetSearchResults

System.GetChatLogMode

Brief

Gets the current display mode of the Chat Log.

Definition

```
ChatLogMode System.GetChatLogMode()
```

Arguments

None

Return Values

The current mode of the Chat Log. This will be one of the following:

- ChatLogMode.Minimized
- ChatLogMode.Maximized
- ChatLogMode.Closed

Description

Returns whether the mode of the Chat Log is minimized (visible but not interactive), maximized (visible and interactive), or closed.

Examples

```
if (System.GetChatLogMode() ~= ChatLogMode.Closed) then
    HideHud()
end
```

See Also

- System.SetChatLogMode

System.GetConsoleTestValue

Brief

Gets the value of the specified test key

Definition

```
string System.GetConsoleTestValue()
```

Arguments

None

Return Values

Description

Returns the value for a specified test key, which can be set (in the HDK and QA/test builds) by using the command 'SetConsoleTest' either

manually through the debug console or through a hubstartup.txt file

Examples

```
[With 'SetConsoleTest MyKey,MyValue' included in hubstartup.txt]
value = System.GetConsoleTestValue( 'MyKey' )
```

See Also

None

System.GetExecTitleErrorCode

Brief

Retrieves the error code set upon ExecTitle failure.

Definition

```
number System.GetExecTitleErrorCode()
```

Arguments

Return Values

error code

Description

Retrieves the error code set upon ExecTitle failure. This function is only valid within the error callback specified as part of the System.ExecTitle() call. Error codes: 1 - unknown error 2 - parental level restricted 3 - exec title cancelled The script should use this opportunity to gracefully recover from the failure.

Examples

```
function LaunchTitle()
            System.ExecTitle( "XXXX12345", gameOptions, "MyLaunchErrorCB()" )
        end

        function MyLaunchErrorCB()
            local errorCode = System.GetExecTitleErrorCode()

            -- Insert script here to handle the error
        end
```

See Also

System.GetNatType

Brief

Gets the NAT type of the user.

Definition

```
int System.GetNatType()
```

Arguments

None

Return Values

The NAT type for the user, which will be either 1, 2 or 3.

Description

The user's NAT type will depend on their network configuration. Users with NAT type 1 and 2 should not have any issues with PS Home functionality. Users on NAT type 3 may not be able to fully join RtGame sessions.

Examples

```
allowUserToHostRtGame = true
if (System.GetNatType() == 3) then
    allowUserToHostRtGame = false
    -- display OSD informing user of the issue
end
```

See Also

None

System.GetRegionsTerritory

Brief

Returns the territory of the specified region.

Definition

```
string System.GetRegionsTerritory( string region )
```

Arguments

region - The region to get the territory for, e.g. "en-GB".

Return Values

The territory of the specified region.

Description

Returns the territory of the specified region. Can be one of the following:
"SCEE" (Europe)
"SCEA" (North America)
"SCEJ" (Japan)
"SCEAsia" (Rest of Asia)
Note: the default territory is set to SCEE - this means that if the region submitted is invalid then SCEE is returned.

Examples

```
currentRegion = LocalPlayer.GetCurrentRegion()
print( 'My territory is ' .. System.GetRegionsTerritory( currentRegion ) )
```

See Also

- LocalPlayer.GetCurrentRegion

- LocalPlayer.GetHomeRegion
- LocalPlayer.GetAccountRegion

System.GetScreenBottomRight

Brief

Returns the x and y pixel values of the bottom right of the screen.

Definition

```
Vector4 System.GetScreenBottomRight()
```

Arguments

None

Return Values

A Vector4 containing the x and y pixel values of the bottom right of the screen.

Description

Returns a Vector4 containing the x and y pixel values of the bottom right of the screen. The value differs between widescreen and 4/3 mode.

Examples

```
Renderer.DrawText2d( renderer, System.GetScreenBottomRight(), "Bottom right" )
```

See Also

- System.IsWideScreen
- System.GetScreenTopLeft

System.GetScreenTopLeft

Brief

Returns the x and y pixel values of the top left of the screen.

Definition

```
Vector4 System.GetScreenTopLeft()
```

Arguments

None

Return Values

A Vector4 containing the x and y pixel values of the top left of the screen.

Description

Returns a Vector4 containing the x and y pixel values of the top left of the screen. The value differs between widescreen and 4/3 mode.

Examples

```
Renderer.DrawText2d( renderer, System.GetScreenTopLeft(), "Top left" )
```

See Also

- System.IsWideScreen
- System.GetScreenBottomRight

System.GetUserValue

Brief

Gets a value for a specified system key

Definition

```
string System.GetUserValue(string key)
```

Arguments

key - Key to get

Return Values

Returns the key's value or NIL if none is set

Description

Gets a system option value. System keys are controlled by HPG, and relate to future systems.

Examples

```

```

See Also

None

System.GetVideoUploadResult

Brief

Retrieve the result of the last video upload request.

Definition

```
Result System.GetVideoUploadResult()
```

Arguments

None

Return Values

One of the following result codes:
VidUploadResult.Success - success

VidUploadResult.ErrBusy - the upload utility was busy
VidUploadResult.ErrNetwork - a network error occurred
VidUploadResult.ErrUnavailable - the upload service is currently unavailable
VidUploadResult.ErrQuote - service limit on upload size was exceeded
VidUploadResult.ErrMemory - out of memory
VidUploadResult.ErrFile - unable to open specified file
VidUploadResult.ErrUnknown - other unspecified error

Description

Reports the current status of the video upload utility.

Examples

```
if ( System.GetVideoUploadResult() ~= VidUploadResult.Success ) then
    print('Upload failed')
end
```

See Also

- System.UploadVideo
- System.GetVideoUploadStatus
- System.GetVideoUploadUrl

System.GetVideoUploadStatus

Brief

Retrieve the status of the current video upload request, if any.

Definition

```
Status System.GetVideoUploadStatus()
```

Arguments

None

Return Values

One of the following status codes:
VidUploadStatus.Idle - no current upload
VidUploadStatus.Busy - upload utility is currently busy with another task and is unavailable
VidUploadStatus.Uploading - currently uploading
VidUploadStatus.Error - an error occurred during upload

Description

Reports the current status of the video upload utility.

Examples

```
while ( System.GetVideoUploadStatus() == VidUploadStatus.Uploading ) do
    coroutine.yield()
end
```

See Also

- System.UploadVideo

- System.GetVideoUploadResult
- System.GetVideoUploadUrl

System.GetVideoUploadUrl

Brief

Retrieve the resulting URL of the last successfully uploaded video.

Definition

```
string System.GetVideoUploadUrl()
```

Arguments

None

Return Values

The URL as a string.

Description

Returns the URL of the uploaded video if upload succeeded, or an empty string if the upload failed.

Examples

```
print( 'Your video has been uploaded to ' .. System.GetVideoUploadUrl() )
```

See Also

- System.UploadVideo
- System.GetVideoUploadStatus
- System.GetVideoUploadResult

System.IsGameLaunchSessionActive

Brief

Query whether the local user is part of a game launch session.

Definition

```
boolean System.IsGameLaunchSessionActive()
```

Arguments

None

Return Values

True if the local user is part of a game launch session, or false otherwise.

Description

If the local user is part of a game launch session, a script may wish to prevent access to certain functionality. This function will indicate whether the user is joined to a game launch session.

Examples

```
if (not System.IsGameLaunchSessionActive()) then
    StartVideoRecording()
end
```

See Also

None

System.IsLookXInverted

Brief

Returns the user setting for X axis look invert option.

Definition

```
boolean System.IsLookXInverted()
```

Arguments

None

Return Values

true if the user's settings in PS Home has the X axis look inverted and false otherwise.

Description

Returns true if the user's settings in PS Home has the X axis look inverted and false otherwise.

Examples

```
if ( System.IsLookXInverted() ) then
    angle = -angle
end
```

See Also

None

System.IsLookYInverted

Brief

Returns the user setting for Y axis look invert option.

Definition

```
boolean System.IsLookYInverted()
```

Arguments

None

Return Values

true if the user's settings in PS Home has the Y axis look inverted and false otherwise.

Description

Returns true if the user's settings in PS Home has the Y axis look inverted and false otherwise.

Examples

```
if ( System.IsLookYInverted() ) then
    angle = -angle
end
```

See Also

None

System.IsVideoSystemLocked

Brief

Returns the status of the video system lock.

Definition

```
boolean System.IsVideoSystemLocked()
```

Arguments

None

Return Values

True if the lock is being used.

Description

This will check to see if any object currently has the video system locked. This function will not return true if the scene is using the video system - this is not considered a lock, only objects will lock the video system.

Examples

```
if ( not System.IsVideoSystemLocked() ) then
    if ( System.VideoSystemLock() ) then
        print("Video system locked!")
    else
        print("Oops, the scene is using the video system. Lock will never succeed.")
    end
else
    print("Oops, another object holds the lock! Can try again later.")
end
```

See Also

- System.VideoSystemLock
- System.VideoSystemUnlock

- Screen.Create

System.IsWideScreen

Brief

Queries whether PS Home is running in widescreen mode or not.

Definition

```
boolean System.IsWideScreen()
```

Arguments

None

Return Values

true if PS Home is running in widescreen mode and false otherwise.

Description

Returns true if PS Home is running in widescreen mode and false otherwise.

Examples

```
if System.IsWideScreen() then
    print( "You are in widescreen" )
end
```

See Also

- System.GetScreenBottomRight
- System.GetScreenTopLeft

System.PostScore

Brief

Posts a score to the score service

Definition

```
System.PostScore( string guid, number score, table metadata = nil )
```

Arguments

guid - A unique developer guid for external trackingscore - The users scoremetadata - An optional lua table containing metadata for this score

Return Values

Description

This function will upload a users score value to the score service with optional metadata. ill be encoded in Json and if any errors occur parsing the data the score will be uploaded without it.

Examples

```
LoadLibrary( "System" )
local score = 100
local data = { track ="Track9", car="Car4", time="1:23:57" }
System.PostScore( "8dd2eea4-fd37-48c1-9f90-d0cb16ca307f", score, data )
```

See Also

- Json.Decode

System.ProfanityFilterText

Brief

Filters a string through the profanity filter.

Definition

```
string System.ProfanityFilterText( string unfilteredText )
```

Arguments

unfilteredText - The string to be filtered for profanity.

Return Values

The string as filtered by the profanity filter.

Description

Filters a string through the profanity filter. To test if filtering happened, perform string equality test on original and returned text. Note that profanity filtering should be applied by the receiver of text (for display purposes) and not the sender.

Examples

```
displayText = System.ProfanityFilterText( receivedText )
```

See Also

None

System.SendNpMessage

Brief

Sends an NP message to the specified users.

Definition

```
System.SendNpMessage( User[] users, string subject, string body )
```

Arguments

users - An array of User objects.subject - The subject of the message.body - The body/main text of the message.

Return Values

None.

Description

Sends an NP message to the specified users. A popup appears requiring user interaction to send the message. Users are grouped in batches of 12. The maximum subject length is 18 bytes including 1 byte for null termination. The maximum text body is 512 bytes including 1 byte for null termination.

Examples

```
friends = LocalPlayer.GetFriends()
subject = "Game Launching"
body = "Hi! We're about to game launch. Come join us!"
System.SendNpMessage( friends, subject, body )
```

See Also

- LocalPlayer.GetFriends

System.SetChatLogMode

Brief

Sets the display mode of the Chat Log.

Definition

```
System.SetChatLogMode(ChatLogMode mode)
```

Arguments

mode - the desired mode of the Chat Log. This will be one of the following:

- ChatLogMode.Minimized
- ChatLogMode.Maximized
- ChatLogMode.Closed

Return Values

Description

Sets whether the mode of the Chat Log is minimized (visible but not interactive), maximized (visible and interactive), or closed.

Examples

```
System.SetChatLogMode(ChatLogMode.Maximized)
```

See Also

- System.GetChatLogMode

System.UploadVideo

Brief

Upload a specified video to a supported video sharing site.

Definition

```
boolean System.UploadVideo( FileHandle file, siteCode, ... )
```

Arguments

file - a handle to a video file obtained from VideoRecorder.GetOutputFilesiteCode - video sharing site ID code (see details)... - further parameters, depending on the site code (see details)

Return Values

true if upload was successfully initiated, false otherwise.

Description

Initiates an upload of the specified video to a supported video sharing site. A variety of extra metadata may be supplied depending on the site, see the list below:

YouTube (R) - [string clientId] [string developerKey] [string title], [string description], [string keyword1], <boolean isPrivate = true> <number rating = 0> <string keyword2 = """>, <string keyword3 = """>
clientId - Client ID chosen by developer, max 63 bytes ASCII alphanumeric characters only.
developerKey - Developer key on YouTube. A developer should obtain this from YouTube directly (see below).
title - Video title, max 60 bytes in UTF-8.
description - Video description, max 1023 bytes in UTF-8.
keyword1, keyword2, keyword3 - Video keywords, each max 24 bytes in UTF-8.
isPrivate - Private video (only visible to uploading user).
rating - Viewer age restriction, 0 indicates no restrictions.

The client ID can be chosen by the developer and represent the object using the upload API. The developer key must be obtained from YouTube (http://code.google.com/apis/youtube/dashboard/) and is separate from the user's username and password. The user will be prompted for their own username and password automatically as part of the upload. Also note that it is forbidden to charge a fee either directly or indirectly to upload videos to YouTube, this is against the YouTube Terms of Use.

[] = compulsory
<> = optional
The siteCodes available are:
VideoSite.YouTube
Note that a successful result from this function only indicates that the upload process has started successfully, you must check the progress using the GetVideoUploadStatus() function. The upload is not complete until the utility status is VidUploadStatus.Idle or VidUploadStatus.Error. If this function returns false, this will indicate the uploader is busy, the script may try again a fixed number of times but it should not assume that the upload will always succeed.

Examples

```
local video = VideoRecorder.GetOutputFile()
if ( System.UploadVideo( video, VideoSite.YouTube, myClientId, myDevKey, 'My Video', 'My PS Home
video', 'Home' ) ) then
    print('Video uploading...')
end
```

See Also

- System.GetVideoUploadStatus
- System.GetVideoUploadResult
- System.GetVideoUploadUrl
- VideoRecorder.GetOutputFile

System.VideoSystemLock

Brief

Allows a non scene object to attempt to gain access to the video system.

Definition

```
boolean System.VideoSystemLock()
```

Arguments

None

Return Values

True if the object successfully locked the video system or false otherwise.

Description

The VideoSystem is primarily used for scenes and scene objects. Other types of object may attempt to lock the video system for use if it is not already in use by the scene. If the lock returns true, this instance of the object may freely create and play videos. Once playback has finished, the object must unlock the video system for use by other objects. If the lock fails, no videos can be created by non scene objects. A script can distinguish between the two lock failure cases (another object having the lock, and the scene using the video system) by checking the IsVideoSystemLocked function; this will only return true in the former case.

Examples

```
local screen = Screen.Create()
if ( not System.IsVideoSystemLocked() ) then
    if ( System.VideoSystemLock() ) then
        print("Video system locked!")
        Screen.SetContent( screen, "http://www.mycontentserver.com/video.mp4" )
    else
        print("Oops, the scene is using the video system. Lock will never succeed.")
    end
else
    print("Oops, another object holds the lock! Can try again later.")
end
```

See Also

- System.VideoSystemUnlock
- System.IsVideoSystemLocked
- Screen.Create

System.VideoSystemUnlock

Brief

Unlocks the video system assuming the object currently has the lock.

Definition

```
System.VideoSystemUnlock()
```

Arguments

None

Return Values

None.

Description

Assuming a non scene object successfully acquired the lock for the video system, this function will release the lock once the object has finished using it. This must be called before termination of the object. The unlock call will only succeed once the video has been deleted so it may not be possible to immediately reacquire the lock after this call.

Examples

```
if ( not Screen.IsContentPlaying( myScreen ) ) then
    System.VideoSystemUnlock()
end
```

See Also

- System.VideoSystemLock
- System.IsVideoSystemLocked
- Screen.Create

System.WebBrowserIsClosed

Brief

Checks if the user has closed the browser.

Definition

```
boolean System.WebBrowserIsClosed()
```

Arguments

None

Return Values

true if the user has closed the browser, false otherwise.

Description

Checks if the user has closed the browser.

Examples

```
local hasFinished = System.WebBrowserIsClosed()
```

See Also

- System.WebBrowserOpen

System.WebBrowserOpen

Brief

Requests opening of the web browser.

Definition

```
number System.WebBrowserOpen( string url )
```

Arguments

url - The HTTP-based URL to open in the web browser.

Return Values

The result of the open request. Return values are:
1 - Browser was opened
0 - Browser is not available yet
-1 - Browser is already running
-2 - Browser is not permitted to run

Description

Checks for the availability of the web browser and opens it at the specified URL if available. The web browser that is opened supports Adobe Flash Player 7.

Examples

```
local browserWasOpened = System.WebBrowserOpen( "http://www.example.com/" )
```

See Also

- System.WebBrowserIsClosed

System.WriteScriptedGameLaunchXML

Brief

Writes XML data from a buffer to the file SCRIPTEDGAMELAUNCHSESSION.XML

Definition

```
System.WriteScriptedGameLaunchXML( memcontainer )
```

Arguments

memcontainer - Memory container containing XML data.

Return Values

None.

Description

Writes XML data from a buffer to the file SCRIPTEDGAMELAUNCHSESSION.XML in the hard drive directory given by the following Cell SDK function: int cellGameGetHomeLaunchOptionPath(char *commonPath, char *personalPath)

Examples

```
local container = MemoryContainer.Create( 2048 )
local writer = BasicGenx.Create( container )

BasicGenx.StartDocument( writer )
-- Write XML data into container here.
BasicGenx.EndDocument( writer )

System.WriteScriptedGameLaunchXML( container )
```

See Also

- BasicGenx.Create
- BasicGenx.EndDocument

Texture

Texture.Find

Brief

Looks up and returns the texture with the specified name.

Definition

```
Texture Texture.Find( string name, ResourcePack pack = nil )
```

Arguments

name - the string name associated with the texture as specified in an object's resources.xml file.pack - ResourcePack to load texture from, if required.

Return Values

The texture instance associated with the specified string name.

Description

Looks up and returns the texture with the specified name. Returns nil if the texture is not found. If a resource pack is specified, the resource pack is searched rather than the local object.

Examples

```
tex = Texture.Find( "myimage" )
tex2 = Texture.Find( "myimage", resourcePack )
```

See Also

- Texture.GetHeight
- Texture.GetWidth

Texture.GetHeight

Brief

Returns the height of the texture.

Definition

```
number Texture.GetHeight( Texture texture )
```

Arguments

texture - The texture to query.

Return Values

The height in pixels of the texture.

Description

Returns the height of the texture.

Examples

```
tex = Texture.Find( "myimage" )
height = Texture.GetHeight( tex )
```

See Also

- Texture.GetWidth

Texture.GetWidth

Brief

Returns the width of the texture.

Definition

```
number Texture.GetWidth( Texture texture )
```

Arguments

texture - The texture to query.

Return Values

The width in pixels of the texture.

Description

Returns the width of the texture.

Examples

```
tex = Texture.Find( "myimage" )
width = Texture.GetWidth( tex )
```

See Also

- Texture.GetHeight

Time

Time.AddDays

Brief

Add a specified number of days to the stored time.

Definition

```
Time.AddDays(Time t, number days)
```

Arguments

t - Time to modify.days - Number of days to add. Negative values will cause subtraction.

Return Values

None.

Description

Adds the specified number of days onto the time stored in the time object. Will correctly account for month lengths and leap years. To subtract, pass a negative value for days. Only integer values are permitted, fractional values will be truncated.

Examples

```
time = Time.Create()
time:SetFromServer()
time:AddDays(90)            -- add 90 days onto the current server time
```

See Also

- Time.AddMinutes
- Time.AddHours
- Time.AddSeconds
- Time.AddWeeks
- Time.AddMonths
- Time.AddYears

Time.AddHours

Brief

Add a specified number of hours to the stored time.

Definition

```
Time.AddHours(Time t, number hours)
```

Arguments

t - Time to modify.hours - Number of hours to add. Negative values will cause subtraction.

Return Values

None.

Description

Adds the specified number of hours onto the time stored in the time object. Will correctly account for month lengths and leap years. To subtract, pass a negative value for hours. Only integer values are permitted, fractional values will be truncated.

Examples

```
time = Time.Create()
time:SetFromServer()
time:AddHours(60)           -- add 60 hours onto the current server time
```

See Also

- Time.AddMinutes
- Time.AddSeconds
- Time.AddDays
- Time.AddWeeks
- Time.AddMonths
- Time.AddYears

Time.AddMinutes

Brief

Add a specified number of minutes to the stored time.

Definition

```
Time.AddMinutes(Time t, number minutes)
```

Arguments

t - Time to modify.minutes - Number of minutes to add. Negative values will cause subtraction.

Return Values

None.

Description

Adds the specified number of minutes onto the time stored in the time object. Will correctly account for month lengths and leap years. To subtract, pass a negative value for minutes.Only integer values are permitted, fractional values will be truncated.

Examples

```
time = Time.Create()
time:SetFromServer()
time:AddMinutes(45)         -- add 45 minutes onto the current server time
```

See Also

- Time.AddSeconds
- Time.AddHours
- Time.AddDays
- Time.AddWeeks
- Time.AddMonths
- Time.AddYears

Time.AddMonths

Brief

Add a specified number of months to the stored time.

Definition

```
Time.AddMonths(Time t, number months)
```

Arguments

t - Time to modify.months - Number of months to add. Negative values will cause subtraction.

Return Values

None.

Description

Adds the specified number of months onto the time stored in the time object. Will correctly account for month lengths and leap years. To subtract, pass a negative value for months. Only integer values are permitted, fractional values will be truncated.

Examples

```
time = Time.Create()
time:SetFromServer()
time:AddMonths(6)           -- add 6 months onto the current server time
```

See Also

- Time.AddMinutes
- Time.AddHours
- Time.AddDays
- Time.AddWeeks
- Time.AddSeconds
- Time.AddYears

Time.AddSeconds

Brief

Add a specified number of seconds to the stored time.

Definition

```
Time.AddSeconds(Time t, number seconds)
```

Arguments

t - Time to modify.seconds - Number of seconds to add. Negative values will cause subtraction.

Return Values

None.

Description

Adds the specified number of seconds onto the time stored in the time object. Will correctly account for month lengths and leap years. To subtract, pass a negative value for seconds. Only integer values are permitted, fractional values will be truncated.

Examples

```
time = Time.Create()
time:SetFromServer()
time:AddSeconds(30)         -- add 30 seconds onto the current server time
```

See Also

- Time.AddMinutes
- Time.AddHours
- Time.AddDays
- Time.AddWeeks
- Time.AddMonths
- Time.AddYears

Time.AddWeeks

Brief

Add a specified number of weeks to the stored time.

Definition

```
Time.AddWeeks(Time t, number weeks)
```

Arguments

t - Time to modify.weeks - Number of weeks to add. Negative values will cause subtraction.

Return Values

None.

Description

Adds the specified number of weeks onto the time stored in the time object. Will correctly account for month lengths and leap years. To subtract, pass a negative value for weeks. Only integer values are permitted, fractional values will be truncated.

Examples

```
time = Time.Create()
time:SetFromServer()
time:AddWeeks(8)            -- add 8 weeks onto the current server time
```

See Also

- Time.AddMinutes
- Time.AddHours
- Time.AddDays
- Time.AddSeconds
- Time.AddMonths
- Time.AddYears

Time.AddYears

Brief

Add a specified number of years to the stored time.

Definition

```
Time.AddYears(Time t, number years)
```

Arguments

t - Time to modify.years - Number of years to add. Negative values will cause subtraction.

Return Values

None.

Description

Adds the specified number of years onto the time stored in the time object. Will correctly account for month lengths and leap years. To subtract, pass a negative value for years. Only integer values are permitted, fractional values will be truncated.

Examples

```
time = Time.Create()
time:SetFromServer()
time:AddYears(1)            -- add 1 year onto the current server time
```

See Also

- Time.AddMinutes
- Time.AddHours
- Time.AddDays
- Time.AddWeeks
- Time.AddMonths
- Time.AddSeconds

Time.ConvertToLocal

Brief

Convert the stored time from UTC time to local time.

Definition

```
Time.ConvertToLocal(Time t)
```

Arguments

t - Time to modify.

Return Values

None.

Description

Converts a time specified in UTC time to local time, based on the current system settings. Note that this relies on the user having set the system timezone settings correctly, and as such should not be relied upon for absolute accuracy.

Examples

```
local t = Time.Create()
t:SetFromServer()
t:ConvertToLocal()
print("Current server time in local time is:", t)
```

See Also

- Time.ConvertToUtc

Time.ConvertToUtc

Brief

Convert the stored time from local time to UTC time.

Definition

```
Time.ConvertToUtc(Time t)
```

Arguments

t - Time to modify.

Return Values

None.

Description

Converts a time specified in local time to UTC time, based on the current system settings. Note that this relies on the user having set the system timezone settings correctly, and as such should not be relied upon for absolute accuracy.

Examples

```
local t = Time.Create()
t:SetFromLocal()
t:ConvertToUtc()
print("Current time in UTC is:", t)
```

See Also

- Time.ConvertToLocal

Time.Copy

Brief

Copy a Time value.

Definition

```
Time.Copy(Time destination, Time source)
```

Arguments

destination - Time value to copy into.source - Time value to copy.

Return Values

None.

Description

Copies a Time value from one instance to another.

Examples

```
local time1 = Time.Create(2011, 10, 31)
local time2 = Time.Create()
Time.Copy(time2, time1)
```

See Also

- Time.Create

Time.Create

Brief

Creates a new Time object.

Definition

```
Time Time.Create()
Time Time.Create(number year, number month, number day, number hours = 0, number minutes = 0, number
seconds = 0)
Time Time.Create(Time t)
```

Arguments

year - Year to set.month - Month to set.day - Day to set.hours - Hours to set (24 hour time).minutes - Minutes to set.seconds - Seconds to set.t - Time to copy.

Return Values

A new Time instance.

Description

Creates a new Time object. By default it is set to 00:00 1/1/1. When defining a specific date/time, invalid date/times will throw a Lua error.

Examples

```
t = Time.Create()
```

See Also

None

Time.GetAsPosixTimestamp

Brief

Retrieve the stored time value as a POSIX-compatible 64-bit timestamp.

Definition

```
Time.GetAsPosixTimestamp(Time t, BigInt<64> timestamp)
```

Arguments

t - Time object to convert.timestamp - A 64-bit BigInt value to receive the timestamp.

Return Values

Description

Retrieve the time value converted to a POSIX-compatible 64-bit timestamp (time_t).

Examples

```
local timestamp = BigInt.Create(64)
local t = Time.Create()
t:SetFromServer(t)
t:GetAsPosixTimestamp(timestamp)
```

See Also

- Time.SetFromPosixTimestamp

Time.GetDate

Brief

Returns a string representing the date.

Definition

```
string Time.GetDate( Time time )
```

Arguments

time - The time object to query.

Return Values

A string representing the date.

Description

Returns a string representing the date following the DDMMYYYY, YYYYMMDD, or MMDDYYYY format specified in the XMB.

Examples

```
t = Time.Create()
Time.SetFromLocal( t )
print( Time.GetTime() )
```

See Also

None

Time.GetDayOfMonth

Brief

Returns the day of the month value.

Definition

```
number Time.GetDayOfMonth( Time time )
```

Arguments

time - The time object to query.

Return Values

The day of the month value, ranging from 1 to 31.

Description

Returns the day of the month value.

Examples

```
t = Time.Create()
Time.SetFromServer( t )
print( 'The server day of the month is ' .. Time.GetDayOfMonth( t ) .. '.' )
```

See Also

None

Time.GetDayOfWeek

Brief

Returns the day of the week value.

Definition

```
number Time.GetDayOfWeek( Time time )
```

Arguments

time - The time object to query.

Return Values

The day of the week value, ranging from 1 (Sunday) to 7 (Saturday).

Description

Returns the day of the week value.

Examples

```
t = Time.Create()
Time.SetFromServer( t )
print( 'The server day of the week is ' .. Time.GetDayOfWeek( t ) .. '.' )
```

See Also

None

Time.GetDifference

Brief

Compute the difference between two Time values in years, months, days, hours, minutes and seconds.

Definition

```
number, number, number, number, number, number Time.GetDifference(Time t1, Time t2)
```

Arguments

t1 - The first time value.t2 - The second time value.

Return Values

The difference from t1 to t2 in terms of years, months, days, hours, minutes and seconds, in that order.

Description

Computes the offset of t2 relative to t1 (i.e. the difference will be negative if t2 is earlier than t1, otherwise will be positive). This function is more efficient than using the other GetDifferenceIn* functions when requiring more than one unit. However, the individual functions are recommended when only a single unit is needed (e.g. days, weeks, etc).

Examples

```
local now = Time.Create()
now:SetFromServer()
local y, m, d, hh, mm, ss = Time.GetDifference(lastSpaceVisit, now)
print(string.format("You were last here %d years, %d months, %d days, %d hours, %d minutes and %d
seconds ago",
    y, m, d, hh, mm, ss))
```

See Also

- Time.GetDifferenceInSeconds
- Time.GetDifferenceInMinutes
- Time.GetDifferenceInHours
- Time.GetDifferenceInMonths
- Time.GetDifferenceInYears

Time.GetDifferenceInDays

Brief

Get the difference between two Time values in days.

Definition

```
number Time.GetDifferenceInDays(Time t1, Time t2)
```

Arguments

t1 - The first time value.t2 - The second time value.

Return Values

The number of whole days between t1 and t2.

Description

Computes the offset of t2 from t1 in days (i.e. the difference will be negative if t2 is earlier than t1, or positive otherwise). If the difference overflows the size of a Lua number, the maximum representable Lua integer will be returned.

Examples

```
local t = Time.Create()
t:SetFromServer()
-- if last visit to the space was over 5 days ago
if (Time.GetDifferenceInDays(lastVisitTime, t) >= 5) then
    print("What kept you?")
end
```

See Also

- Time.GetDifferenceInSeconds
- Time.GetDifferenceInMinutes
- Time.GetDifferenceInHours
- Time.GetDifferenceInYears

- Time.GetDifference

Time.GetDifferenceInHours

Brief

Get the difference between two Time values in hours.

Definition

```
number Time.GetDifferenceInHours(Time t1, Time t2)
```

Arguments

t1 - The first time value.t2 - The second time value.

Return Values

The number of whole hours between t1 and t2.

Description

Computes the offset of t2 from t1 in hours (i.e. the difference will be negative if t2 is earlier than t1, or positive otherwise). If the difference overflows the size of a Lua number, the maximum representable Lua integer will be returned.

Examples

```
local t = Time.Create()
t:SetFromServer()
-- if last server update was over 2 hours ago, upload again
if (Time.GetDifferenceInHours(lastServerUpdate, t) >= 2) then
    UploadToServer()
end
```

See Also

- Time.GetDifferenceInSeconds
- Time.GetDifferenceInMinutes
- Time.GetDifferenceInDays
- Time.GetDifferenceInYears
- Time.GetDifference

Time.GetDifferenceInMinutes

Brief

Get the difference between two Time values in minutes.

Definition

```
number Time.GetDifferenceInMinutes(Time t1, Time t2)
```

Arguments

t1 - The first time value.t2 - The second time value.

Return Values

The number of whole minutes between t1 and t2.

Description

Computes the offset of t2 from t1 in minutes (i.e. the difference will be negative if t2 is earlier than t1, or positive otherwise). If the difference overflows the size of a Lua number, the maximum representable Lua integer will be returned.

Examples

```
local t = Time.Create()
t:SetFromServer()
-- if last server update was over 30 minutes ago, upload again
if (Time.GetDifferenceInMinutes(lastServerUpdate, t) >= 30) then
    UploadToServer()
end
```

See Also

- Time.GetDifferenceInSeconds
- Time.GetDifferenceInHours
- Time.GetDifferenceInDays
- Time.GetDifferenceInYears
- Time.GetDifference

Time.GetDifferenceInMonths

Brief

Get the difference between two Time values in months.

Definition

```
number Time.GetDifferenceInMonths(Time t1, Time t2)
```

Arguments

t1 - The first time value.t2 - The second time value.

Return Values

The number of whole months between t1 and t2.

Description

Computes the offset of t2 from t1 in months (i.e. the difference will be negative if t2 is earlier than t1, or positive otherwise). If the difference overflows the size of a Lua number, the maximum representable Lua integer will be returned.

Examples

```
local t = Time.Create()
t:SetFromServer()
-- if last visit to the space was over 3 months ago
if (Time.GetDifferenceInMonths(lastVisitTime, t) >= 3) then
    print("What kept you?")
end
```

See Also

- Time.GetDifferenceInSeconds
- Time.GetDifferenceInMinutes
- Time.GetDifferenceInHours
- Time.GetDifferenceInYears
- Time.GetDifference

Time.GetDifferenceInSeconds

Brief

Get the difference between two Time values in seconds.

Definition

```
number Time.GetDifferenceInSeconds(Time t1, Time t2)
```

Arguments

t1 - The first time value.t2 - The second time value.

Return Values

The number of whole seconds between t1 and t2.

Description

Computes the offset of t2 from t1 in seconds (i.e. the difference will be negative if t2 is earlier than t1, or positive otherwise). If the difference overflows the size of a Lua number, the maximum representable Lua integer will be returned.

Examples

```
local t = Time.Create()
t:SetFromServer()
-- if last server update was over 2.5 minutes ago, upload again
if (Time.GetDifferenceInSeconds(lastServerUpdate, t) >= 150) then
    UploadToServer()
end
```

See Also

- Time.GetDifferenceInMinutes
- Time.GetDifferenceInHours
- Time.GetDifferenceInDays
- Time.GetDifferenceInYears
- Time.GetDifference

Time.GetDifferenceInWeeks

Brief

Get the difference between two Time values in weeks.

Definition

```
number Time.GetDifferenceInWeeks(Time t1, Time t2)
```

Arguments

t1 - The first time value.t2 - The second time value.

Return Values

The number of whole weeks between t1 and t2.

Description

Computes the offset of t2 from t1 in weeks (i.e. the difference will be negative if t2 is earlier than t1, or positive otherwise). If the difference overflows the size of a Lua number, the maximum representable Lua integer will be returned.

Examples

```
local t = Time.Create()
t:SetFromServer()
-- if last visit to the space was over 12 weeks ago
if (Time.GetDifferenceInWeeks(lastVisitTime, t) >= 12) then
    print("What kept you?")
end
```

See Also

- Time.GetDifferenceInSeconds
- Time.GetDifferenceInMinutes
- Time.GetDifferenceInHours
- Time.GetDifferenceInYears
- Time.GetDifference

Time.GetDifferenceInYears

Brief

Get the difference between two Time values in years.

Definition

```
number Time.GetDifferenceInYears(Time t1, Time t2)
```

Arguments

t1 - The first time value.t2 - The second time value.

Return Values

The number of whole years between t1 and t2.

Description

Computes the offset of t2 from t1 in years (i.e. the difference will be negative if t2 is earlier than t1, or positive otherwise). If the difference overflows the size of a Lua number, the maximum representable Lua integer will be returned.

Examples

```
local t = Time.Create()
t:SetFromServer()
-- if last visit to the space was over 2 years ago
if (Time.GetDifferenceInYears(lastVisitTime, t) >= 2) then
    print("What kept you?")
end
```

See Also

- Time.GetDifferenceInSeconds
- Time.GetDifferenceInMinutes
- Time.GetDifferenceInHours
- Time.GetDifferenceInMonths
- Time.GetDifference

Time.GetHour

Brief

Returns the hour value.

Definition

```
number Time.GetHour( Time time )
```

Arguments

time - The time object to query.

Return Values

The hour value, ranging from 0 to 23.

Description

Returns the hour value.

Examples

```
t = Time.Create()
Time.SetFromServer( t )
print( 'The server hour is ' .. Time.GetHour( t ) .. '.' )
```

See Also

None

Time.GetMinute

Brief

Returns the minute value.

Definition

```
number Time.GetMinute( Time time )
```

Arguments

time - The time object to query.

Return Values

The minute value, ranging from 0 to 59.

Description

Returns the minute value.

Examples

```
t = Time.Create()
Time.SetFromServer( t )
print( 'The server minute is ' .. Time.GetMinute( t ) .. '.' )
```

See Also

None

Time.GetMonth

Brief

Returns the month value.

Definition

```
number Time.GetMonth( Time time )
```

Arguments

time - The time object to query.

Return Values

The month value, ranging from 1 to 12.

Description

Returns the month value.

Examples

```
t = Time.Create()
Time.SetFromServer( t )
print( 'The server month is ' .. Time.GetMonth( t ) .. '.' )
```

See Also

None

Time.GetSecond

Brief

Returns the second value.

Definition

```
number Time.GetSecond( Time time )
```

Arguments

time - The time object to query.

Return Values

The second value, ranging from 0 to 59.

Description

Returns the second value.

Examples

```
t = Time.Create()
Time.SetFromServer( t )
print( 'The server second is ' .. Time.GetSecond( t ) .. '.' )
```

See Also

None

Time.GetTime

Brief

Returns a string representing the time.

Definition

```
string Time.GetTime( Time time )
```

Arguments

time - The time object to query.

Return Values

A string representing the time.

Description

Returns a string representing the time following the 12h or 24h format specified in the XMB.

Examples

```
t = Time.Create()
Time.SetFromLocal( t )
print( Time.GetTime() )
```

See Also

None

Time.GetYear

Brief

Returns the year value.

Definition

```
number Time.GetYear( Time time )
```

Arguments

time - The time object to query.

Return Values

The year value, ranging from 0 to 9999.

Description

Returns the year value.

Examples

```
t = Time.Create()
Time.SetFromServer( t )
print( 'The server year is ' .. Time.GetYear( t ) .. '.' )
```

See Also

None

Time.IsValid

Brief

Check if a stored date/time value represents a valid date and time.

Definition

```
boolean Time.IsValid(Time time)
```

Arguments

time - The Time object to check.

Return Values

True if the date and time contain valid values, or false if not.

Description

Since 1.70 the Time object will no longer internally enforce that the stored value is always valid. The validity will only be checked if attempting to perform arithmetic (Add* or GetDifference* functions) or convert the time between time zones or formats. If you wish to check a time for validity without generating a Lua error this function can be used, and will check that the date/time value represents a valid date and time, taking account of leap years where appropriate.

Examples

```
if (not myTime:IsValid()) then
    myTime:SetFromServer()  -- revert to a valid time
end
```

See Also

None

Time.SetDayOfMonth

Brief

Sets the day of the month value.

Definition

```
Time.SetDayOfMonth( Time time, number value )
```

Arguments

time - The time object to update.value - The day of the month value, ranging from 1 to 31.

Return Values

None

Description

Sets the day of the month value.

Examples

```
t = Time.Create()
Time.SetDayOfMonth( t, 29 )
print( 'The server day of the month is ' .. Time.GetDayOfMonth( t ) .. '.' )
```

See Also

None

Time.SetFromLocal

Brief

Sets the time object to the current local time of the PlayStation(R)3.

Definition

```
boolean Time.SetFromLocal( Time time )
```

Arguments

time - The time object to update.

Return Values

true if successful and false otherwise.

Description

Sets the time object to the current local time of the PlayStation(R)3. This is the time the user has set their PlayStation(R)3 to.

Examples

```
t = Time.Create()
Time.SetFromLocal( t )
print( 'The time is ' .. Time.GetTime( t ) .. '.' )
```

See Also

None

Time.SetFromPosixTimestamp

Brief

Convert a POSIX-compatible 64-bit timestamp into a Time value.

Definition

```
Time.SetFromPosixTimestamp(Time t, BigInt<64> timestamp)
```

Arguments

t - Time object to modify.timestamp - A 64-bit BigInt value containing the timestamp.

Return Values

Description

Converts a POSIX-compatible 64-bit timestamp (time_t) into a Time value.

Examples

```
local timestamp = BigInt.Create(64)
ReadTimestampFromExternalSource(timestamp)
local t = Time.Create()
t:SetFromPosixTimestamp(timestamp)
```

See Also

- Time.GetAsPosixTimestamp

Time.SetFromServer

Brief

Sets the time object to the current server time.

Definition

```
boolean Time.SetFromServer( Time time )
```

Arguments

time - The time object to update.

Return Values

true if successful and false otherwise.

Description

Sets the time object to the current server time. The time object is not automatically updated with the server time so you must call it every time you want to refresh it. The server time is independent of the PlayStation(R)3 region or local time settings and always returns UTC (Coordinated Universal Time). Note that the Home client does not need to be run in online mode for this function to work. However, the target machine must be signed in to the PSN.

Examples

```
t = Time.Create()
Time.SetFromServer( t )
print( 'The time is ' .. Time.GetTime( t ) .. '.' )
```

See Also

None

Time.SetHour

Brief

Sets the hour value.

Definition

```
Time.SetHour( Time time, number value )
```

Arguments

time - The time object to update.value - The hour value, ranging from 0 to 23.

Return Values

None

Description

Sets the hour value.

Examples

```
t = Time.Create()
Time.SetHour( t, 12 )
print( 'The server hour is ' .. Time.GetHour( t ) .. '.' )
```

See Also

None

Time.SetMinute

Brief

Sets the minute value.

Definition

```
Time.SetMinute( Time time, number value )
```

Arguments

time - The time object to update.value - The minute value, ranging from 0 to 59.

Return Values

None

Description

Sets the minute value.

Examples

```
t = Time.Create()
Time.SetMinute( t, 55 )
print( 'The server minute is ' .. Time.GetMinute( t ) .. '.' )
```

See Also

None

Time.SetMonth

Brief

Sets the month value.

Definition

```
Time.SetMonth( Time time, number value )
```

Arguments

time - The time object to update.value - The month value, ranging from 1 to 12.

Return Values

None

Description

Sets the month value.

Examples

```
t = Time.Create()
Time.SetMonth( t, 12 )
print( 'The server month is ' .. Time.GetMonth( t ) .. '.' )
```

See Also

None

Time.SetSecond

Brief

Sets the second value.

Definition

```
Time.SetSecond( Time time, number value )
```

Arguments

time - The time object to update.value - The second value, ranging from 0 to 59.

Return Values

None

Description

Sets the second value.

Examples

```
t = Time.Create()
Time.SetSecond( t, 12 )
print( 'The server second is ' .. Time.GetSecond( t ) .. '.' )
```

See Also

None

Time.SetYear

Brief

Sets the year value.

Definition

```
Time.SetYear( Time time, number value )
```

Arguments

time - The time object to update.value - The year value, ranging from 0 to 9999.

Return Values

None

Description

Sets the year value.

Examples

```
t = Time.Create()
Time.SetYear( t, 1066 )
print( 'The server year is ' .. Time.GetYear( t ) .. '.' )
```

See Also

None

TriggerVolume

TriggerVolume.AddCallback

Brief

Add a callback function to the trigger volume.

Definition

```
TriggerVolume.AddCallback( TriggerVolume triggerVolume, function callback )
```

Arguments

triggerVolume - The trigger volume to update.callback - The callback function to add.

Return Values

None.

Description

Add a callback function to the trigger volume. The callback will be called when an object enters or exits the trigger volume. The callback should be of the form:
Callback(TriggerVolume triggerVolume, number overlapState, number numOverlaps) Valid values for overlapState are:
TriggerVolumeOverlapState.Enter
TriggerVolumeOverlapState.Exit The callback will be called once per-frame, per-overlap-state if there are any overlaps of that state during that frame.

Examples

```
function MyTriggerVolumeCallback1( triggerVolume, state, count )
    if state == TriggerVolumeOverlapState.Enter then
        print( count .. " new overlaps for trigger volume " .. tostring( triggerVolume ) .. " this
frame" )
    elseif state == TriggerVolumeOverlapState.Exit then
        print( count .. " old overlaps for trigger volume " .. tostring( triggerVolume ) .. " this
frame" )
    end
end

function MyTriggerVolumeCallback2( triggerVolume, state, count )
    for i = 1, count do
        local   overlapType = TriggerVolume.GetOverlapType( triggerVolume, i )
        local   overlapData = TriggerVolume.GetOverlapData( triggerVolume, i )
        print( "trigger volume " .. tostring( triggerVolume ) .. " overlap " .. tostring( i ) .. ":
type=" .. tostring( overlapType ) .. ", object=" .. tostring( overlapData.object ) )
    end
end

triggerVolumes = {}
triggerVolumes[1] = Collision.CreateTriggerVolume( "simpleTriggerVolume1", 16,
TriggerVolumeShape.Sphere, 1.0 )
triggerVolumes[2] = Collision.CreateTriggerVolume( "simpleTriggerVolume2", 16, TriggerVolumeShape.Box,
3.0, 2.0, 4.0 )
triggerVolumes[3] = Collision.CreateTriggerVolume( "resourceTriggerVolume1", 16,
"kTriggerVolumeShapeResource1" )
triggerVolumes[4] = Collision.CreateTriggerVolume( "resourceTriggerVolume2", 16, Resource.Find(
"kTriggerVolumeShapeResource2" ) )

TriggerVolume.SetLocalMatrix( triggerVolumes[1], triggerVolumeTransformMatrix[1] )
TriggerVolume.SetLocalMatrix( triggerVolumes[2], TriggerVolume.GetLocalMatrix( triggerVolumes[1] ) )
TriggerVolume.SetPosition( triggerVolumes[3], triggerVolumePosition[1] )
TriggerVolume.SetPosition( triggerVolumes[4], TriggerVolume.GetPosition( triggerVolumes[3] ) )

for i = 1, #triggerVolumes do
    TriggerVolume.AddCallback( triggerVolumes[i], MyTriggerVolumeCallback1 )
    TriggerVolume.AddCallback( triggerVolumes[i], MyTriggerVolumeCallback2 )
    if TriggerVolume.IsLayerOverlapsEnabled( triggerVolumes[i], CollisionLayer.Scene ) then
        TriggerVolume.EnableLayerOverlaps( triggerVolumes[i], CollisionLayer.Scene, false )
    end
    if not TriggerVolume.IsEnabled( triggerVolumes[i] ) then
        TriggerVolume.Enable( triggerVolumes[i], true )
    end
end

TriggerVolume.RemoveCallback( triggerVolumes[3], MyTriggerVolumeCallback2 )
TriggerVolume.RemoveAllCallbacks( triggerVolumes[4] )

local   numOverlaps = TriggerVolume.GetOverlapCount( triggerVolumes[1] )

for i = 1, numOverlaps do
    local   overlapType = TriggerVolume.GetOverlapType( triggerVolumes[1], i )
    local   overlapData = TriggerVolume.GetOverlapData( triggerVolumes[1], i )
    print( "trigger volume " .. tostring( triggerVolumes[1] ) .. " overlap " .. tostring( i ) .. ":
type=" .. tostring( overlapType ) .. ", object=" .. tostring( overlapData.object ) )
end
```

See Also

None

TriggerVolume.Enable

Brief

Enable/disable the trigger volume.

Definition

```
TriggerVolume.Enable( TriggerVolume triggerVolume, boolean enabled = true, Phys3dWorld world = nil )
```

Arguments

triggerVolume - The trigger volume to update.enabled - true to enable the trigger volume, false otherwise.world - The physics world to add the trigger volume to.

Return Values

None.

Description

Enable/disable the trigger volume. A trigger volume must be enabled to detect overlaps. If the world parameter is not specified (or nil), the default physics world will be used.

Examples

```
function MyTriggerVolumeCallback1( triggerVolume, state, count )
    if state == TriggerVolumeOverlapState.Enter then
        print( count .. " new overlaps for trigger volume " .. tostring( triggerVolume ) .. " this
frame" )
    elseif state == TriggerVolumeOverlapState.Exit then
        print( count .. " old overlaps for trigger volume " .. tostring( triggerVolume ) .. " this
frame" )
    end
end

function MyTriggerVolumeCallback2( triggerVolume, state, count )
    for i = 1, count do
        local   overlapType = TriggerVolume.GetOverlapType( triggerVolume, i )
        local   overlapData = TriggerVolume.GetOverlapData( triggerVolume, i )
        print( "trigger volume " .. tostring( triggerVolume ) .. " overlap " .. tostring( i ) .. ":
type=" .. tostring( overlapType ) .. ", object=" .. tostring( overlapData.object ) )
    end
end

triggerVolumes = {}
triggerVolumes[1] = Collision.CreateTriggerVolume( "simpleTriggerVolume1", 16,
TriggerVolumeShape.Sphere, 1.0 )
triggerVolumes[2] = Collision.CreateTriggerVolume( "simpleTriggerVolume2", 16, TriggerVolumeShape.Box,
3.0, 2.0, 4.0 )
triggerVolumes[3] = Collision.CreateTriggerVolume( "resourceTriggerVolume1", 16,
"kTriggerVolumeShapeResource1" )
triggerVolumes[4] = Collision.CreateTriggerVolume( "resourceTriggerVolume2", 16, Resource.Find(
"kTriggerVolumeShapeResource2" ) )

TriggerVolume.SetLocalMatrix( triggerVolumes[1], triggerVolumeTransformMatrix[1] )
TriggerVolume.SetLocalMatrix( triggerVolumes[2], TriggerVolume.GetLocalMatrix( triggerVolumes[1] ) )
TriggerVolume.SetPosition( triggerVolumes[3], triggerVolumePosition[1] )
TriggerVolume.SetPosition( triggerVolumes[4], TriggerVolume.GetPosition( triggerVolumes[3] ) )

for i = 1, #triggerVolumes do
    TriggerVolume.AddCallback( triggerVolumes[i], MyTriggerVolumeCallback1 )
    TriggerVolume.AddCallback( triggerVolumes[i], MyTriggerVolumeCallback2 )
    if TriggerVolume.IsLayerOverlapsEnabled( triggerVolumes[i], CollisionLayer.Scene ) then
        TriggerVolume.EnableLayerOverlaps( triggerVolumes[i], CollisionLayer.Scene, false )
    end
    if not TriggerVolume.IsEnabled( triggerVolumes[i] ) then
        TriggerVolume.Enable( triggerVolumes[i], true )
    end
end

TriggerVolume.RemoveCallback( triggerVolumes[3], MyTriggerVolumeCallback2 )
TriggerVolume.RemoveAllCallbacks( triggerVolumes[4] )

local   numOverlaps = TriggerVolume.GetOverlapCount( triggerVolumes[1] )

for i = 1, numOverlaps do
    local   overlapType = TriggerVolume.GetOverlapType( triggerVolumes[1], i )
    local   overlapData = TriggerVolume.GetOverlapData( triggerVolumes[1], i )
    print( "trigger volume " .. tostring( triggerVolumes[1] ) .. " overlap " .. tostring( i ) .. ":
type=" .. tostring( overlapType ) .. ", object=" .. tostring( overlapData.object ) )
end
```

See Also

None

TriggerVolume.EnableDebugRender

Brief

Enable debug rendering of the trigger volume.

Definition

```
TriggerVolume.EnableDebugRender( TriggerVolume triggerVolume, boolean enable )
```

Arguments

triggerVolume - The trigger volume to update.enable - true to enable, false otherwise.

Return Values

None.

Description

Enable debug rendering of the trigger volume. The trigger volume shape will be rendered:

1. red - enabled, with overlaps
2. green - enabled, no overlaps
3. blue - disabled

Examples

See Also

None

TriggerVolume.EnableLayerOverlaps

Brief

Enable/disable overlap detection between the trigger volume and objects of a given collision layer.

Definition

```
TriggerVolume.EnableLayerOverlaps( TriggerVolume triggerVolume, enum collisionLayer, boolean enabled =
true )
```

Arguments

triggerVolume - The trigger volume to update.collisionLayer - The CollisionLayer enum.enabled - true to enable overlaps, false otherwise.

Return Values

None.

Description

Enable/disable overlap detection between the trigger volume and objects of a given collision layer. Changes will only take effect for new overlaps. If the trigger volume and an object are already overlapping (or very close), disabling their overlap detection via this function will not take effect until they separate. Also, if the trigger volume and an object are overlapping but overlap detection is disabled, then enabling their overlap detection via this function will not take effect until they separate. Therefore, it is advised to set up layer overlap detection before enabling the trigger volume. In other words, call TriggerVolume.EnableLayerOverlaps before TriggerVolume.Enable. The following CollisionLayer enums are supported by trigger volumes: * CollisionLayer.Avatar - just detects the local player, not remote players * CollisionLayer.Entity - detects entities and placed furniture * CollisionLayer.FurniturePlacement - detects furniture during the furniture placement process

Examples

```
function MyTriggerVolumeCallback1( triggerVolume, state, count )
    if state == TriggerVolumeOverlapState.Enter then
        print( count .. " new overlaps for trigger volume " .. tostring( triggerVolume ) .. " this
frame" )
    elseif state == TriggerVolumeOverlapState.Exit then
        print( count .. " old overlaps for trigger volume " .. tostring( triggerVolume ) .. " this
frame" )
    end
end

function MyTriggerVolumeCallback2( triggerVolume, state, count )
    for i = 1, count do
        local   overlapType = TriggerVolume.GetOverlapType( triggerVolume, i )
        local   overlapData = TriggerVolume.GetOverlapData( triggerVolume, i )
        print( "trigger volume " .. tostring( triggerVolume ) .. " overlap " .. tostring( i ) .. ":
type=" .. tostring( overlapType ) .. ", object=" .. tostring( overlapData.object ) )
    end
end

triggerVolumes = {}
triggerVolumes[1] = Collision.CreateTriggerVolume( "simpleTriggerVolume1", 16,
TriggerVolumeShape.Sphere, 1.0 )
triggerVolumes[2] = Collision.CreateTriggerVolume( "simpleTriggerVolume2", 16, TriggerVolumeShape.Box,
3.0, 2.0, 4.0 )
triggerVolumes[3] = Collision.CreateTriggerVolume( "resourceTriggerVolume1", 16,
"kTriggerVolumeShapeResource1" )
triggerVolumes[4] = Collision.CreateTriggerVolume( "resourceTriggerVolume2", 16, Resource.Find(
"kTriggerVolumeShapeResource2" ) )

TriggerVolume.SetLocalMatrix( triggerVolumes[1], triggerVolumeTransformMatrix[1] )
TriggerVolume.SetLocalMatrix( triggerVolumes[2], TriggerVolume.GetLocalMatrix( triggerVolumes[1] ) )
TriggerVolume.SetPosition( triggerVolumes[3], triggerVolumePosition[1] )
TriggerVolume.SetPosition( triggerVolumes[4], TriggerVolume.GetPosition( triggerVolumes[3] ) )

for i = 1, #triggerVolumes do
    TriggerVolume.AddCallback( triggerVolumes[i], MyTriggerVolumeCallback1 )
    TriggerVolume.AddCallback( triggerVolumes[i], MyTriggerVolumeCallback2 )
    if TriggerVolume.IsLayerOverlapsEnabled( triggerVolumes[i], CollisionLayer.Scene ) then
        TriggerVolume.EnableLayerOverlaps( triggerVolumes[i], CollisionLayer.Scene, false )
    end
    if not TriggerVolume.IsEnabled( triggerVolumes[i] ) then
        TriggerVolume.Enable( triggerVolumes[i], true )
    end
end

TriggerVolume.RemoveCallback( triggerVolumes[3], MyTriggerVolumeCallback2 )
TriggerVolume.RemoveAllCallbacks( triggerVolumes[4] )

local   numOverlaps = TriggerVolume.GetOverlapCount( triggerVolumes[1] )

for i = 1, numOverlaps do
    local   overlapType = TriggerVolume.GetOverlapType( triggerVolumes[1], i )
    local   overlapData = TriggerVolume.GetOverlapData( triggerVolumes[1], i )
    print( "trigger volume " .. tostring( triggerVolumes[1] ) .. " overlap " .. tostring( i ) .. ":
type=" .. tostring( overlapType ) .. ", object=" .. tostring( overlapData.object ) )
end
```

See Also

None

TriggerVolume.GetLocalMatrix

Brief

Gets the trigger volume's transform matrix.

Definition

```
TriggerVolume.GetLocalMatrix( TriggerVolume triggerVolume, Matrix44 matrix )
Matrix44 TriggerVolume.GetLocalMatrix( TriggerVolume triggerVolume )
```

Arguments

triggerVolume - The trigger volume to update.matrix - The variable to store the trigger volume's transform matrix in.

Return Values

The transform matrix (or nil).

Description

Gets the trigger volume's transform matrix.

Examples

```
function MyTriggerVolumeCallback1( triggerVolume, state, count )
    if state == TriggerVolumeOverlapState.Enter then
        print( count .. " new overlaps for trigger volume " .. tostring( triggerVolume ) .. " this
frame" )
    elseif state == TriggerVolumeOverlapState.Exit then
        print( count .. " old overlaps for trigger volume " .. tostring( triggerVolume ) .. " this
frame" )
    end
end

function MyTriggerVolumeCallback2( triggerVolume, state, count )
    for i = 1, count do
        local   overlapType = TriggerVolume.GetOverlapType( triggerVolume, i )
        local   overlapData = TriggerVolume.GetOverlapData( triggerVolume, i )
        print( "trigger volume " .. tostring( triggerVolume ) .. " overlap " .. tostring( i ) .. ":
type=" .. tostring( overlapType ) .. ", object=" .. tostring( overlapData.object ) )
    end
end

triggerVolumes = {}
triggerVolumes[1] = Collision.CreateTriggerVolume( "simpleTriggerVolume1", 16,
TriggerVolumeShape.Sphere, 1.0 )
triggerVolumes[2] = Collision.CreateTriggerVolume( "simpleTriggerVolume2", 16, TriggerVolumeShape.Box,
3.0, 2.0, 4.0 )
triggerVolumes[3] = Collision.CreateTriggerVolume( "resourceTriggerVolume1", 16,
"kTriggerVolumeShapeResource1" )
triggerVolumes[4] = Collision.CreateTriggerVolume( "resourceTriggerVolume2", 16, Resource.Find(
"kTriggerVolumeShapeResource2" ) )

TriggerVolume.SetLocalMatrix( triggerVolumes[1], triggerVolumeTransformMatrix[1] )
TriggerVolume.SetLocalMatrix( triggerVolumes[2], TriggerVolume.GetLocalMatrix( triggerVolumes[1] ) )
TriggerVolume.SetPosition( triggerVolumes[3], triggerVolumePosition[1] )
TriggerVolume.SetPosition( triggerVolumes[4], TriggerVolume.GetPosition( triggerVolumes[3] ) )

for i = 1, #triggerVolumes do
    TriggerVolume.AddCallback( triggerVolumes[i], MyTriggerVolumeCallback1 )
    TriggerVolume.AddCallback( triggerVolumes[i], MyTriggerVolumeCallback2 )
    if TriggerVolume.IsLayerOverlapsEnabled( triggerVolumes[i], CollisionLayer.Scene ) then
        TriggerVolume.EnableLayerOverlaps( triggerVolumes[i], CollisionLayer.Scene, false )
    end
    if not TriggerVolume.IsEnabled( triggerVolumes[i] ) then
        TriggerVolume.Enable( triggerVolumes[i], true )
    end
end

TriggerVolume.RemoveCallback( triggerVolumes[3], MyTriggerVolumeCallback2 )
TriggerVolume.RemoveAllCallbacks( triggerVolumes[4] )

local   numOverlaps = TriggerVolume.GetOverlapCount( triggerVolumes[1] )

for i = 1, numOverlaps do
    local   overlapType = TriggerVolume.GetOverlapType( triggerVolumes[1], i )
    local   overlapData = TriggerVolume.GetOverlapData( triggerVolumes[1], i )
    print( "trigger volume " .. tostring( triggerVolumes[1] ) .. " overlap " .. tostring( i ) .. ":
type=" .. tostring( overlapType ) .. ", object=" .. tostring( overlapData.object ) )
end
```

See Also

None

TriggerVolume.GetOverlapCount

Brief

Get the current number of overlaps.

Definition

```
number TriggerVolume.GetOverlapCount( TriggerVolume triggerVolume )
```

Arguments

triggerVolume - The trigger volume to query.

Return Values

The number of overlaps.

Description

Get the current number of overlaps.

Examples

```
function MyTriggerVolumeCallback1( triggerVolume, state, count )
    if state == TriggerVolumeOverlapState.Enter then
        print( count .. " new overlaps for trigger volume " .. tostring( triggerVolume ) .. " this
frame" )
    elseif state == TriggerVolumeOverlapState.Exit then
        print( count .. " old overlaps for trigger volume " .. tostring( triggerVolume ) .. " this
frame" )
    end
end

function MyTriggerVolumeCallback2( triggerVolume, state, count )
    for i = 1, count do
        local   overlapType = TriggerVolume.GetOverlapType( triggerVolume, i )
        local   overlapData = TriggerVolume.GetOverlapData( triggerVolume, i )
        print( "trigger volume " .. tostring( triggerVolume ) .. " overlap " .. tostring( i ) .. ":
type=" .. tostring( overlapType ) .. ", object=" .. tostring( overlapData.object ) )
    end
end

triggerVolumes = {}
triggerVolumes[1] = Collision.CreateTriggerVolume( "simpleTriggerVolume1", 16,
TriggerVolumeShape.Sphere, 1.0 )
triggerVolumes[2] = Collision.CreateTriggerVolume( "simpleTriggerVolume2", 16, TriggerVolumeShape.Box,
3.0, 2.0, 4.0 )
triggerVolumes[3] = Collision.CreateTriggerVolume( "resourceTriggerVolume1", 16,
"kTriggerVolumeShapeResource1" )
triggerVolumes[4] = Collision.CreateTriggerVolume( "resourceTriggerVolume2", 16, Resource.Find(
"kTriggerVolumeShapeResource2" ) )

TriggerVolume.SetLocalMatrix( triggerVolumes[1], triggerVolumeTransformMatrix[1] )
TriggerVolume.SetLocalMatrix( triggerVolumes[2], TriggerVolume.GetLocalMatrix( triggerVolumes[1] ) )
TriggerVolume.SetPosition( triggerVolumes[3], triggerVolumePosition[1] )
TriggerVolume.SetPosition( triggerVolumes[4], TriggerVolume.GetPosition( triggerVolumes[3] ) )

for i = 1, #triggerVolumes do
    TriggerVolume.AddCallback( triggerVolumes[i], MyTriggerVolumeCallback1 )
    TriggerVolume.AddCallback( triggerVolumes[i], MyTriggerVolumeCallback2 )
    if TriggerVolume.IsLayerOverlapsEnabled( triggerVolumes[i], CollisionLayer.Scene ) then
        TriggerVolume.EnableLayerOverlaps( triggerVolumes[i], CollisionLayer.Scene, false )
    end
    if not TriggerVolume.IsEnabled( triggerVolumes[i] ) then
        TriggerVolume.Enable( triggerVolumes[i], true )
    end
end

TriggerVolume.RemoveCallback( triggerVolumes[3], MyTriggerVolumeCallback2 )
TriggerVolume.RemoveAllCallbacks( triggerVolumes[4] )

local   numOverlaps = TriggerVolume.GetOverlapCount( triggerVolumes[1] )

for i = 1, numOverlaps do
    local   overlapType = TriggerVolume.GetOverlapType( triggerVolumes[1], i )
    local   overlapData = TriggerVolume.GetOverlapData( triggerVolumes[1], i )
    print( "trigger volume " .. tostring( triggerVolumes[1] ) .. " overlap " .. tostring( i ) .. ":
type=" .. tostring( overlapType ) .. ", object=" .. tostring( overlapData.object ) )
end
```

See Also

None

TriggerVolume.GetOverlapData

Brief

Get the data relating to a trigger volume overlap.

Definition

```
table TriggerVolume.GetOverlapData( TriggerVolume triggerVolume, number index )
```

Arguments

triggerVolume - The trigger volume to query.index - The index of the overlap.

Return Values

A table of data or nil

Description

Get the data relating to a trigger volume overlap. Only to be called from a trigger volume callback. The table will contain members:
type - a TriggerVolumeOverlapType value
object - the type of this data depends on the type of the overlap Valid TriggerVolumeOverlapType values are:
TriggerVolumeOverlapType.Entity
TriggerVolumeOverlapType.Person
TriggerVolumeOverlapType.Furniture

Examples

```
function MyTriggerVolumeCallback1( triggerVolume, state, count )
    if state == TriggerVolumeOverlapState.Enter then
        print( count .. " new overlaps for trigger volume " .. tostring( triggerVolume ) .. " this
frame" )
    elseif state == TriggerVolumeOverlapState.Exit then
        print( count .. " old overlaps for trigger volume " .. tostring( triggerVolume ) .. " this
frame" )
    end
end

function MyTriggerVolumeCallback2( triggerVolume, state, count )
    for i = 1, count do
        local   overlapType = TriggerVolume.GetOverlapType( triggerVolume, i )
        local   overlapData = TriggerVolume.GetOverlapData( triggerVolume, i )
        print( "trigger volume " .. tostring( triggerVolume ) .. " overlap " .. tostring( i ) .. ":
type=" .. tostring( overlapType ) .. ", object=" .. tostring( overlapData.object ) )
    end
end

triggerVolumes = {}
triggerVolumes[1] = Collision.CreateTriggerVolume( "simpleTriggerVolume1", 16,
TriggerVolumeShape.Sphere, 1.0 )
triggerVolumes[2] = Collision.CreateTriggerVolume( "simpleTriggerVolume2", 16, TriggerVolumeShape.Box,
3.0, 2.0, 4.0 )
triggerVolumes[3] = Collision.CreateTriggerVolume( "resourceTriggerVolume1", 16,
"kTriggerVolumeShapeResource1" )
triggerVolumes[4] = Collision.CreateTriggerVolume( "resourceTriggerVolume2", 16, Resource.Find(
"kTriggerVolumeShapeResource2" ) )

TriggerVolume.SetLocalMatrix( triggerVolumes[1], triggerVolumeTransformMatrix[1] )
TriggerVolume.SetLocalMatrix( triggerVolumes[2], TriggerVolume.GetLocalMatrix( triggerVolumes[1] ) )
TriggerVolume.SetPosition( triggerVolumes[3], triggerVolumePosition[1] )
TriggerVolume.SetPosition( triggerVolumes[4], TriggerVolume.GetPosition( triggerVolumes[3] ) )

for i = 1, #triggerVolumes do
    TriggerVolume.AddCallback( triggerVolumes[i], MyTriggerVolumeCallback1 )
    TriggerVolume.AddCallback( triggerVolumes[i], MyTriggerVolumeCallback2 )
    if TriggerVolume.IsLayerOverlapsEnabled( triggerVolumes[i], CollisionLayer.Scene ) then
        TriggerVolume.EnableLayerOverlaps( triggerVolumes[i], CollisionLayer.Scene, false )
    end
    if not TriggerVolume.IsEnabled( triggerVolumes[i] ) then
        TriggerVolume.Enable( triggerVolumes[i], true )
    end
end

TriggerVolume.RemoveCallback( triggerVolumes[3], MyTriggerVolumeCallback2 )
TriggerVolume.RemoveAllCallbacks( triggerVolumes[4] )

local   numOverlaps = TriggerVolume.GetOverlapCount( triggerVolumes[1] )

for i = 1, numOverlaps do
    local   overlapType = TriggerVolume.GetOverlapType( triggerVolumes[1], i )
    local   overlapData = TriggerVolume.GetOverlapData( triggerVolumes[1], i )
    print( "trigger volume " .. tostring( triggerVolumes[1] ) .. " overlap " .. tostring( i ) .. ":
type=" .. tostring( overlapType ) .. ", object=" .. tostring( overlapData.object ) )
end
```

See Also

None

TriggerVolume.GetOverlapType

Brief

Get the type of a trigger volume overlap.

Definition

```
enum TriggerVolume.GetOverlapType( TriggerVolume triggerVolume, number index )
```

Arguments

triggerVolume - The trigger volume to query.index - The index of the overlap.

Return Values

A TriggerVolumeOverlapType enum or nil.

Description

Get the type of a trigger volume overlap. Only to be called from a trigger volume callback. Valid TriggerVolumeOverlapType values are:
TriggerVolumeOverlapType.Entity
TriggerVolumeOverlapType.Person
TriggerVolumeOverlapType.Furniture

Examples

```
function MyTriggerVolumeCallback1( triggerVolume, state, count )
    if state == TriggerVolumeOverlapState.Enter then
        print( count .. " new overlaps for trigger volume " .. tostring( triggerVolume ) .. " this
frame" )
    elseif state == TriggerVolumeOverlapState.Exit then
        print( count .. " old overlaps for trigger volume " .. tostring( triggerVolume ) .. " this
frame" )
    end
end

function MyTriggerVolumeCallback2( triggerVolume, state, count )
    for i = 1, count do
        local   overlapType = TriggerVolume.GetOverlapType( triggerVolume, i )
        local   overlapData = TriggerVolume.GetOverlapData( triggerVolume, i )
        print( "trigger volume " .. tostring( triggerVolume ) .. " overlap " .. tostring( i ) .. ":
type=" .. tostring( overlapType ) .. ", object=" .. tostring( overlapData.object ) )
    end
end

triggerVolumes = {}
triggerVolumes[1] = Collision.CreateTriggerVolume( "simpleTriggerVolume1", 16,
TriggerVolumeShape.Sphere, 1.0 )
triggerVolumes[2] = Collision.CreateTriggerVolume( "simpleTriggerVolume2", 16, TriggerVolumeShape.Box,
3.0, 2.0, 4.0 )
triggerVolumes[3] = Collision.CreateTriggerVolume( "resourceTriggerVolume1", 16,
"kTriggerVolumeShapeResource1" )
triggerVolumes[4] = Collision.CreateTriggerVolume( "resourceTriggerVolume2", 16, Resource.Find(
"kTriggerVolumeShapeResource2" ) )

TriggerVolume.SetLocalMatrix( triggerVolumes[1], triggerVolumeTransformMatrix[1] )
TriggerVolume.SetLocalMatrix( triggerVolumes[2], TriggerVolume.GetLocalMatrix( triggerVolumes[1] ) )
TriggerVolume.SetPosition( triggerVolumes[3], triggerVolumePosition[1] )
TriggerVolume.SetPosition( triggerVolumes[4], TriggerVolume.GetPosition( triggerVolumes[3] ) )

for i = 1, #triggerVolumes do
    TriggerVolume.AddCallback( triggerVolumes[i], MyTriggerVolumeCallback1 )
    TriggerVolume.AddCallback( triggerVolumes[i], MyTriggerVolumeCallback2 )
    if TriggerVolume.IsLayerOverlapsEnabled( triggerVolumes[i], CollisionLayer.Scene ) then
        TriggerVolume.EnableLayerOverlaps( triggerVolumes[i], CollisionLayer.Scene, false )
    end
    if not TriggerVolume.IsEnabled( triggerVolumes[i] ) then
        TriggerVolume.Enable( triggerVolumes[i], true )
    end
end

TriggerVolume.RemoveCallback( triggerVolumes[3], MyTriggerVolumeCallback2 )
TriggerVolume.RemoveAllCallbacks( triggerVolumes[4] )

local   numOverlaps = TriggerVolume.GetOverlapCount( triggerVolumes[1] )

for i = 1, numOverlaps do
    local   overlapType = TriggerVolume.GetOverlapType( triggerVolumes[1], i )
    local   overlapData = TriggerVolume.GetOverlapData( triggerVolumes[1], i )
    print( "trigger volume " .. tostring( triggerVolumes[1] ) .. " overlap " .. tostring( i ) .. ":
type=" .. tostring( overlapType ) .. ", object=" .. tostring( overlapData.object ) )
end
```

See Also

None

TriggerVolume.GetPosition

Brief

Gets the trigger volume's position.

Definition

```
TriggerVolume.GetPosition( TriggerVolume triggerVolume, Vector4 position )
Vector4 TriggerVolume.GetPosition( TriggerVolume triggerVolume )
```

Arguments

triggerVolume - The trigger volume to update.position - The variable to store the trigger volume's position in.

Return Values

The position vector (or nil).

Description

Gets the trigger volume's position.

Examples

```
function MyTriggerVolumeCallback1( triggerVolume, state, count )
    if state == TriggerVolumeOverlapState.Enter then
        print( count .. " new overlaps for trigger volume " .. tostring( triggerVolume ) .. " this
frame" )
    elseif state == TriggerVolumeOverlapState.Exit then
        print( count .. " old overlaps for trigger volume " .. tostring( triggerVolume ) .. " this
frame" )
    end
end

function MyTriggerVolumeCallback2( triggerVolume, state, count )
    for i = 1, count do
        local   overlapType = TriggerVolume.GetOverlapType( triggerVolume, i )
        local   overlapData = TriggerVolume.GetOverlapData( triggerVolume, i )
        print( "trigger volume " .. tostring( triggerVolume ) .. " overlap " .. tostring( i ) .. ":
type=" .. tostring( overlapType ) .. ", object=" .. tostring( overlapData.object ) )
    end
end

triggerVolumes = {}
triggerVolumes[1] = Collision.CreateTriggerVolume( "simpleTriggerVolume1", 16,
TriggerVolumeShape.Sphere, 1.0 )
triggerVolumes[2] = Collision.CreateTriggerVolume( "simpleTriggerVolume2", 16, TriggerVolumeShape.Box,
3.0, 2.0, 4.0 )
triggerVolumes[3] = Collision.CreateTriggerVolume( "resourceTriggerVolume1", 16,
"kTriggerVolumeShapeResource1" )
triggerVolumes[4] = Collision.CreateTriggerVolume( "resourceTriggerVolume2", 16, Resource.Find(
"kTriggerVolumeShapeResource2" ) )

TriggerVolume.SetLocalMatrix( triggerVolumes[1], triggerVolumeTransformMatrix[1] )
TriggerVolume.SetLocalMatrix( triggerVolumes[2], TriggerVolume.GetLocalMatrix( triggerVolumes[1] ) )
TriggerVolume.SetPosition( triggerVolumes[3], triggerVolumePosition[1] )
TriggerVolume.SetPosition( triggerVolumes[4], TriggerVolume.GetPosition( triggerVolumes[3] ) )

for i = 1, #triggerVolumes do
    TriggerVolume.AddCallback( triggerVolumes[i], MyTriggerVolumeCallback1 )
    TriggerVolume.AddCallback( triggerVolumes[i], MyTriggerVolumeCallback2 )
    if TriggerVolume.IsLayerOverlapsEnabled( triggerVolumes[i], CollisionLayer.Scene ) then
        TriggerVolume.EnableLayerOverlaps( triggerVolumes[i], CollisionLayer.Scene, false )
    end
    if not TriggerVolume.IsEnabled( triggerVolumes[i] ) then
        TriggerVolume.Enable( triggerVolumes[i], true )
    end
end

TriggerVolume.RemoveCallback( triggerVolumes[3], MyTriggerVolumeCallback2 )
TriggerVolume.RemoveAllCallbacks( triggerVolumes[4] )

local   numOverlaps = TriggerVolume.GetOverlapCount( triggerVolumes[1] )

for i = 1, numOverlaps do
    local   overlapType = TriggerVolume.GetOverlapType( triggerVolumes[1], i )
    local   overlapData = TriggerVolume.GetOverlapData( triggerVolumes[1], i )
    print( "trigger volume " .. tostring( triggerVolumes[1] ) .. " overlap " .. tostring( i ) .. ":
type=" .. tostring( overlapType ) .. ", object=" .. tostring( overlapData.object ) )
end
```

See Also

None

TriggerVolume.IsEnabled

Brief

Query whether or not the trigger volume is enabled.

Definition

```
boolean TriggerVolume.IsEnabled( TriggerVolume triggerVolume, Phys3dWorld world = nil )
```

Arguments

triggerVolume - The trigger volume to query.world - The physics world.

Return Values

true if enabled, false otherwise.

Description

Query whether or not the trigger volume is enabled. A trigger volume must be enabled to detect overlaps. If the world parameter is specified (and not or nil), the function will only return true if the trigger volume is enabled and added to the specified physics world.

Examples

```
function MyTriggerVolumeCallback1( triggerVolume, state, count )
    if state == TriggerVolumeOverlapState.Enter then
        print( count .. " new overlaps for trigger volume " .. tostring( triggerVolume ) .. " this
frame" )
    elseif state == TriggerVolumeOverlapState.Exit then
        print( count .. " old overlaps for trigger volume " .. tostring( triggerVolume ) .. " this
frame" )
    end
end

function MyTriggerVolumeCallback2( triggerVolume, state, count )
    for i = 1, count do
        local   overlapType = TriggerVolume.GetOverlapType( triggerVolume, i )
        local   overlapData = TriggerVolume.GetOverlapData( triggerVolume, i )
        print( "trigger volume " .. tostring( triggerVolume ) .. " overlap " .. tostring( i ) .. ":
type=" .. tostring( overlapType ) .. ", object=" .. tostring( overlapData.object ) )
    end
end

triggerVolumes = {}
triggerVolumes[1] = Collision.CreateTriggerVolume( "simpleTriggerVolume1", 16,
TriggerVolumeShape.Sphere, 1.0 )
triggerVolumes[2] = Collision.CreateTriggerVolume( "simpleTriggerVolume2", 16, TriggerVolumeShape.Box,
3.0, 2.0, 4.0 )
triggerVolumes[3] = Collision.CreateTriggerVolume( "resourceTriggerVolume1", 16,
"kTriggerVolumeShapeResource1" )
triggerVolumes[4] = Collision.CreateTriggerVolume( "resourceTriggerVolume2", 16, Resource.Find(
"kTriggerVolumeShapeResource2" ) )

TriggerVolume.SetLocalMatrix( triggerVolumes[1], triggerVolumeTransformMatrix[1] )
TriggerVolume.SetLocalMatrix( triggerVolumes[2], TriggerVolume.GetLocalMatrix( triggerVolumes[1] ) )
TriggerVolume.SetPosition( triggerVolumes[3], triggerVolumePosition[1] )
TriggerVolume.SetPosition( triggerVolumes[4], TriggerVolume.GetPosition( triggerVolumes[3] ) )

for i = 1, #triggerVolumes do
    TriggerVolume.AddCallback( triggerVolumes[i], MyTriggerVolumeCallback1 )
    TriggerVolume.AddCallback( triggerVolumes[i], MyTriggerVolumeCallback2 )
    if TriggerVolume.IsLayerOverlapsEnabled( triggerVolumes[i], CollisionLayer.Scene ) then
        TriggerVolume.EnableLayerOverlaps( triggerVolumes[i], CollisionLayer.Scene, false )
    end
    if not TriggerVolume.IsEnabled( triggerVolumes[i] ) then
        TriggerVolume.Enable( triggerVolumes[i], true )
    end
end

TriggerVolume.RemoveCallback( triggerVolumes[3], MyTriggerVolumeCallback2 )
TriggerVolume.RemoveAllCallbacks( triggerVolumes[4] )

local   numOverlaps = TriggerVolume.GetOverlapCount( triggerVolumes[1] )

for i = 1, numOverlaps do
    local   overlapType = TriggerVolume.GetOverlapType( triggerVolumes[1], i )
    local   overlapData = TriggerVolume.GetOverlapData( triggerVolumes[1], i )
    print( "trigger volume " .. tostring( triggerVolumes[1] ) .. " overlap " .. tostring( i ) .. ":
type=" .. tostring( overlapType ) .. ", object=" .. tostring( overlapData.object ) )
end
```

See Also

None

TriggerVolume.IsLayerOverlapsEnabled

Brief

Query whether overlaps between the trigger volume and objects of a given collision layer are enabled.

Definition

```
boolean TriggerVolume.IsLayerOverlapsEnabled( TriggerVolume triggerVolume, enum collisionLayer )
```

Arguments

triggerVolume - The trigger volume to query.collisionLayer - The CollisionLayer enum.

Return Values

true if enabled, false otherwise.

Description

Query whether overlaps between the trigger volume and objects of a given collision layer are enabled.

Examples

```
function MyTriggerVolumeCallback1( triggerVolume, state, count )
    if state == TriggerVolumeOverlapState.Enter then
        print( count .. " new overlaps for trigger volume " .. tostring( triggerVolume ) .. " this
frame" )
    elseif state == TriggerVolumeOverlapState.Exit then
        print( count .. " old overlaps for trigger volume " .. tostring( triggerVolume ) .. " this
frame" )
    end
end

function MyTriggerVolumeCallback2( triggerVolume, state, count )
    for i = 1, count do
        local   overlapType = TriggerVolume.GetOverlapType( triggerVolume, i )
        local   overlapData = TriggerVolume.GetOverlapData( triggerVolume, i )
        print( "trigger volume " .. tostring( triggerVolume ) .. " overlap " .. tostring( i ) .. ":
type=" .. tostring( overlapType ) .. ", object=" .. tostring( overlapData.object ) )
    end
end

triggerVolumes = {}
triggerVolumes[1] = Collision.CreateTriggerVolume( "simpleTriggerVolume1", 16,
TriggerVolumeShape.Sphere, 1.0 )
triggerVolumes[2] = Collision.CreateTriggerVolume( "simpleTriggerVolume2", 16, TriggerVolumeShape.Box,
3.0, 2.0, 4.0 )
triggerVolumes[3] = Collision.CreateTriggerVolume( "resourceTriggerVolume1", 16,
"kTriggerVolumeShapeResource1" )
triggerVolumes[4] = Collision.CreateTriggerVolume( "resourceTriggerVolume2", 16, Resource.Find(
"kTriggerVolumeShapeResource2" ) )

TriggerVolume.SetLocalMatrix( triggerVolumes[1], triggerVolumeTransformMatrix[1] )
TriggerVolume.SetLocalMatrix( triggerVolumes[2], TriggerVolume.GetLocalMatrix( triggerVolumes[1] ) )
TriggerVolume.SetPosition( triggerVolumes[3], triggerVolumePosition[1] )
TriggerVolume.SetPosition( triggerVolumes[4], TriggerVolume.GetPosition( triggerVolumes[3] ) )

for i = 1, #triggerVolumes do
    TriggerVolume.AddCallback( triggerVolumes[i], MyTriggerVolumeCallback1 )
    TriggerVolume.AddCallback( triggerVolumes[i], MyTriggerVolumeCallback2 )
    if TriggerVolume.IsLayerOverlapsEnabled( triggerVolumes[i], CollisionLayer.Scene ) then
        TriggerVolume.EnableLayerOverlaps( triggerVolumes[i], CollisionLayer.Scene, false )
    end
    if not TriggerVolume.IsEnabled( triggerVolumes[i] ) then
        TriggerVolume.Enable( triggerVolumes[i], true )
    end
end

TriggerVolume.RemoveCallback( triggerVolumes[3], MyTriggerVolumeCallback2 )
TriggerVolume.RemoveAllCallbacks( triggerVolumes[4] )

local   numOverlaps = TriggerVolume.GetOverlapCount( triggerVolumes[1] )

for i = 1, numOverlaps do
    local   overlapType = TriggerVolume.GetOverlapType( triggerVolumes[1], i )
    local   overlapData = TriggerVolume.GetOverlapData( triggerVolumes[1], i )
    print( "trigger volume " .. tostring( triggerVolumes[1] ) .. " overlap " .. tostring( i ) .. ":
type=" .. tostring( overlapType ) .. ", object=" .. tostring( overlapData.object ) )
end
```

See Also

None

TriggerVolume.RemoveAllCallbacks

Brief

Remove all callback functions from the trigger volume.

Definition

```
TriggerVolume.RemoveAllCallbacks( TriggerVolume triggerVolume )
```

Arguments

triggerVolume - The trigger volume to update.

Return Values

None.

Description

Remove all callback functions from the trigger volume. Will only remove callbacks added by this script.

Examples

```
function MyTriggerVolumeCallback1( triggerVolume, state, count )
    if state == TriggerVolumeOverlapState.Enter then
        print( count .. " new overlaps for trigger volume " .. tostring( triggerVolume ) .. " this
frame" )
    elseif state == TriggerVolumeOverlapState.Exit then
        print( count .. " old overlaps for trigger volume " .. tostring( triggerVolume ) .. " this
frame" )
    end
end

function MyTriggerVolumeCallback2( triggerVolume, state, count )
    for i = 1, count do
        local   overlapType = TriggerVolume.GetOverlapType( triggerVolume, i )
        local   overlapData = TriggerVolume.GetOverlapData( triggerVolume, i )
        print( "trigger volume " .. tostring( triggerVolume ) .. " overlap " .. tostring( i ) .. ":
type=" .. tostring( overlapType ) .. ", object=" .. tostring( overlapData.object ) )
    end
end

triggerVolumes = {}
triggerVolumes[1] = Collision.CreateTriggerVolume( "simpleTriggerVolume1", 16,
TriggerVolumeShape.Sphere, 1.0 )
triggerVolumes[2] = Collision.CreateTriggerVolume( "simpleTriggerVolume2", 16, TriggerVolumeShape.Box,
3.0, 2.0, 4.0 )
triggerVolumes[3] = Collision.CreateTriggerVolume( "resourceTriggerVolume1", 16,
"kTriggerVolumeShapeResource1" )
triggerVolumes[4] = Collision.CreateTriggerVolume( "resourceTriggerVolume2", 16, Resource.Find(
"kTriggerVolumeShapeResource2" ) )

TriggerVolume.SetLocalMatrix( triggerVolumes[1], triggerVolumeTransformMatrix[1] )
TriggerVolume.SetLocalMatrix( triggerVolumes[2], TriggerVolume.GetLocalMatrix( triggerVolumes[1] ) )
TriggerVolume.SetPosition( triggerVolumes[3], triggerVolumePosition[1] )
TriggerVolume.SetPosition( triggerVolumes[4], TriggerVolume.GetPosition( triggerVolumes[3] ) )

for i = 1, #triggerVolumes do
    TriggerVolume.AddCallback( triggerVolumes[i], MyTriggerVolumeCallback1 )
    TriggerVolume.AddCallback( triggerVolumes[i], MyTriggerVolumeCallback2 )
    if TriggerVolume.IsLayerOverlapsEnabled( triggerVolumes[i], CollisionLayer.Scene ) then
        TriggerVolume.EnableLayerOverlaps( triggerVolumes[i], CollisionLayer.Scene, false )
    end
    if not TriggerVolume.IsEnabled( triggerVolumes[i] ) then
        TriggerVolume.Enable( triggerVolumes[i], true )
    end
end

TriggerVolume.RemoveCallback( triggerVolumes[3], MyTriggerVolumeCallback2 )
TriggerVolume.RemoveAllCallbacks( triggerVolumes[4] )

local   numOverlaps = TriggerVolume.GetOverlapCount( triggerVolumes[1] )

for i = 1, numOverlaps do
    local   overlapType = TriggerVolume.GetOverlapType( triggerVolumes[1], i )
    local   overlapData = TriggerVolume.GetOverlapData( triggerVolumes[1], i )
    print( "trigger volume " .. tostring( triggerVolumes[1] ) .. " overlap " .. tostring( i ) .. ":
type=" .. tostring( overlapType ) .. ", object=" .. tostring( overlapData.object ) )
end
```

See Also

None

TriggerVolume.RemoveCallback

Brief

Remove a callback function from the trigger volume.

Definition

```
TriggerVolume.RemoveCallback( TriggerVolume triggerVolume, function callback )
```

Arguments

triggerVolume - The trigger volume to update.callback - The callback function to remove.

Return Values

None.

Description

Remove a callback function from the trigger volume.

Examples

```
function MyTriggerVolumeCallback1( triggerVolume, state, count )
    if state == TriggerVolumeOverlapState.Enter then
        print( count .. " new overlaps for trigger volume " .. tostring( triggerVolume ) .. " this
frame" )
    elseif state == TriggerVolumeOverlapState.Exit then
        print( count .. " old overlaps for trigger volume " .. tostring( triggerVolume ) .. " this
frame" )
    end
end

function MyTriggerVolumeCallback2( triggerVolume, state, count )
    for i = 1, count do
        local   overlapType = TriggerVolume.GetOverlapType( triggerVolume, i )
        local   overlapData = TriggerVolume.GetOverlapData( triggerVolume, i )
        print( "trigger volume " .. tostring( triggerVolume ) .. " overlap " .. tostring( i ) .. ":
type=" .. tostring( overlapType ) .. ", object=" .. tostring( overlapData.object ) )
    end
end

triggerVolumes = {}
triggerVolumes[1] = Collision.CreateTriggerVolume( "simpleTriggerVolume1", 16,
TriggerVolumeShape.Sphere, 1.0 )
triggerVolumes[2] = Collision.CreateTriggerVolume( "simpleTriggerVolume2", 16, TriggerVolumeShape.Box,
3.0, 2.0, 4.0 )
triggerVolumes[3] = Collision.CreateTriggerVolume( "resourceTriggerVolume1", 16,
"kTriggerVolumeShapeResource1" )
triggerVolumes[4] = Collision.CreateTriggerVolume( "resourceTriggerVolume2", 16, Resource.Find(
"kTriggerVolumeShapeResource2" ) )

TriggerVolume.SetLocalMatrix( triggerVolumes[1], triggerVolumeTransformMatrix[1] )
TriggerVolume.SetLocalMatrix( triggerVolumes[2], TriggerVolume.GetLocalMatrix( triggerVolumes[1] ) )
TriggerVolume.SetPosition( triggerVolumes[3], triggerVolumePosition[1] )
TriggerVolume.SetPosition( triggerVolumes[4], TriggerVolume.GetPosition( triggerVolumes[3] ) )

for i = 1, #triggerVolumes do
    TriggerVolume.AddCallback( triggerVolumes[i], MyTriggerVolumeCallback1 )
    TriggerVolume.AddCallback( triggerVolumes[i], MyTriggerVolumeCallback2 )
    if TriggerVolume.IsLayerOverlapsEnabled( triggerVolumes[i], CollisionLayer.Scene ) then
        TriggerVolume.EnableLayerOverlaps( triggerVolumes[i], CollisionLayer.Scene, false )
    end
    if not TriggerVolume.IsEnabled( triggerVolumes[i] ) then
        TriggerVolume.Enable( triggerVolumes[i], true )
    end
end

TriggerVolume.RemoveCallback( triggerVolumes[3], MyTriggerVolumeCallback2 )
TriggerVolume.RemoveAllCallbacks( triggerVolumes[4] )

local   numOverlaps = TriggerVolume.GetOverlapCount( triggerVolumes[1] )

for i = 1, numOverlaps do
    local   overlapType = TriggerVolume.GetOverlapType( triggerVolumes[1], i )
    local   overlapData = TriggerVolume.GetOverlapData( triggerVolumes[1], i )
    print( "trigger volume " .. tostring( triggerVolumes[1] ) .. " overlap " .. tostring( i ) .. ":
type=" .. tostring( overlapType ) .. ", object=" .. tostring( overlapData.object ) )
end
```

See Also

None

TriggerVolume.SetLocalMatrix

Brief

Sets the trigger volume's local transformation matrix.

Definition

```
TriggerVolume.SetLocalMatrix( TriggerVolume triggerVolume, Matrix44 matrix )
```

Arguments

triggerVolume - The trigger volume to update.matrix - The matrix to set the trigger volume's transform to.

Return Values

None.

Description

Sets the trigger volume's local transformation matrix. The specified matrix is relative to the coordinate system of the trigger volume's parent, if one exists. If the trigger volume has no parent, the matrix is relative to the world space coordinate system. A trigger volume's transform matrix can not support scale.

Examples

```
function MyTriggerVolumeCallback1( triggerVolume, state, count )
    if state == TriggerVolumeOverlapState.Enter then
        print( count .. " new overlaps for trigger volume " .. tostring( triggerVolume ) .. " this
frame" )
    elseif state == TriggerVolumeOverlapState.Exit then
        print( count .. " old overlaps for trigger volume " .. tostring( triggerVolume ) .. " this
frame" )
    end
end

function MyTriggerVolumeCallback2( triggerVolume, state, count )
    for i = 1, count do
        local   overlapType = TriggerVolume.GetOverlapType( triggerVolume, i )
        local   overlapData = TriggerVolume.GetOverlapData( triggerVolume, i )
        print( "trigger volume " .. tostring( triggerVolume ) .. " overlap " .. tostring( i ) .. ":
type=" .. tostring( overlapType ) .. ", object=" .. tostring( overlapData.object ) )
    end
end

triggerVolumes = {}
triggerVolumes[1] = Collision.CreateTriggerVolume( "simpleTriggerVolume1", 16,
TriggerVolumeShape.Sphere, 1.0 )
triggerVolumes[2] = Collision.CreateTriggerVolume( "simpleTriggerVolume2", 16, TriggerVolumeShape.Box,
3.0, 2.0, 4.0 )
triggerVolumes[3] = Collision.CreateTriggerVolume( "resourceTriggerVolume1", 16,
"kTriggerVolumeShapeResource1" )
triggerVolumes[4] = Collision.CreateTriggerVolume( "resourceTriggerVolume2", 16, Resource.Find(
"kTriggerVolumeShapeResource2" ) )

TriggerVolume.SetLocalMatrix( triggerVolumes[1], triggerVolumeTransformMatrix[1] )
TriggerVolume.SetLocalMatrix( triggerVolumes[2], TriggerVolume.GetLocalMatrix( triggerVolumes[1] ) )
TriggerVolume.SetPosition( triggerVolumes[3], triggerVolumePosition[1] )
TriggerVolume.SetPosition( triggerVolumes[4], TriggerVolume.GetPosition( triggerVolumes[3] ) )

for i = 1, #triggerVolumes do
    TriggerVolume.AddCallback( triggerVolumes[i], MyTriggerVolumeCallback1 )
    TriggerVolume.AddCallback( triggerVolumes[i], MyTriggerVolumeCallback2 )
    if TriggerVolume.IsLayerOverlapsEnabled( triggerVolumes[i], CollisionLayer.Scene ) then
        TriggerVolume.EnableLayerOverlaps( triggerVolumes[i], CollisionLayer.Scene, false )
    end
    if not TriggerVolume.IsEnabled( triggerVolumes[i] ) then
        TriggerVolume.Enable( triggerVolumes[i], true )
    end
end

TriggerVolume.RemoveCallback( triggerVolumes[3], MyTriggerVolumeCallback2 )
TriggerVolume.RemoveAllCallbacks( triggerVolumes[4] )

local   numOverlaps = TriggerVolume.GetOverlapCount( triggerVolumes[1] )

for i = 1, numOverlaps do
    local   overlapType = TriggerVolume.GetOverlapType( triggerVolumes[1], i )
    local   overlapData = TriggerVolume.GetOverlapData( triggerVolumes[1], i )
    print( "trigger volume " .. tostring( triggerVolumes[1] ) .. " overlap " .. tostring( i ) .. ":
type=" .. tostring( overlapType ) .. ", object=" .. tostring( overlapData.object ) )
end
```

See Also

None

TriggerVolume.SetPosition

Brief

Sets the trigger volume's position.

Definition

```
TriggerVolume.SetPosition( TriggerVolume triggerVolume, Vector4 position )
```

Arguments

triggerVolume - The trigger volume to update.position - The position to set the trigger volume to.

Return Values

None.

Description

Sets the trigger volume's position. The specified position is relative to the coordinate system of the trigger volume's parent, if one exists. If the trigger volume has no parent, the position is relative to the world space coordinate system.

Examples

```
function MyTriggerVolumeCallback1( triggerVolume, state, count )
    if state == TriggerVolumeOverlapState.Enter then
        print( count .. " new overlaps for trigger volume " .. tostring( triggerVolume ) .. " this
frame" )
    elseif state == TriggerVolumeOverlapState.Exit then
        print( count .. " old overlaps for trigger volume " .. tostring( triggerVolume ) .. " this
frame" )
    end
end

function MyTriggerVolumeCallback2( triggerVolume, state, count )
    for i = 1, count do
        local   overlapType = TriggerVolume.GetOverlapType( triggerVolume, i )
        local   overlapData = TriggerVolume.GetOverlapData( triggerVolume, i )
        print( "trigger volume " .. tostring( triggerVolume ) .. " overlap " .. tostring( i ) .. ":
type=" .. tostring( overlapType ) .. ", object=" .. tostring( overlapData.object ) )
    end
end

triggerVolumes = {}
triggerVolumes[1] = Collision.CreateTriggerVolume( "simpleTriggerVolume1", 16,
TriggerVolumeShape.Sphere, 1.0 )
triggerVolumes[2] = Collision.CreateTriggerVolume( "simpleTriggerVolume2", 16, TriggerVolumeShape.Box,
3.0, 2.0, 4.0 )
triggerVolumes[3] = Collision.CreateTriggerVolume( "resourceTriggerVolume1", 16,
"kTriggerVolumeShapeResource1" )
triggerVolumes[4] = Collision.CreateTriggerVolume( "resourceTriggerVolume2", 16, Resource.Find(
"kTriggerVolumeShapeResource2" ) )

TriggerVolume.SetLocalMatrix( triggerVolumes[1], triggerVolumeTransformMatrix[1] )
TriggerVolume.SetLocalMatrix( triggerVolumes[2], TriggerVolume.GetLocalMatrix( triggerVolumes[1] ) )
TriggerVolume.SetPosition( triggerVolumes[3], triggerVolumePosition[1] )
TriggerVolume.SetPosition( triggerVolumes[4], TriggerVolume.GetPosition( triggerVolumes[3] ) )

for i = 1, #triggerVolumes do
    TriggerVolume.AddCallback( triggerVolumes[i], MyTriggerVolumeCallback1 )
    TriggerVolume.AddCallback( triggerVolumes[i], MyTriggerVolumeCallback2 )
    if TriggerVolume.IsLayerOverlapsEnabled( triggerVolumes[i], CollisionLayer.Scene ) then
        TriggerVolume.EnableLayerOverlaps( triggerVolumes[i], CollisionLayer.Scene, false )
    end
    if not TriggerVolume.IsEnabled( triggerVolumes[i] ) then
        TriggerVolume.Enable( triggerVolumes[i], true )
    end
end

TriggerVolume.RemoveCallback( triggerVolumes[3], MyTriggerVolumeCallback2 )
TriggerVolume.RemoveAllCallbacks( triggerVolumes[4] )

local   numOverlaps = TriggerVolume.GetOverlapCount( triggerVolumes[1] )

for i = 1, numOverlaps do
    local   overlapType = TriggerVolume.GetOverlapType( triggerVolumes[1], i )
    local   overlapData = TriggerVolume.GetOverlapData( triggerVolumes[1], i )
    print( "trigger volume " .. tostring( triggerVolumes[1] ) .. " overlap " .. tostring( i ) .. ":
type=" .. tostring( overlapType ) .. ", object=" .. tostring( overlapData.object ) )
end
```

See Also

None

User

User.GetName

Brief

Returns the user's Online ID.

Definition

```
string User.GetName( User User )
```

Arguments

user - The user object.

Return Values

The user's Online ID.

Description

Returns the user's Online ID, but only if you launch HomeDeveloper.self in online mode. If launching in offline mode, this function returns the string "Me".

Examples

```
myUser = LocalPlayer.GetUser()
print( "My name is " .. User.GetName( myUser ) )
```

See Also

- LocalPlayer.GetUser

User.GetPerson

Brief

Returns a Person object if the user is found in the current instance and nil otherwise.

Definition

```
Person User.GetPerson( User user )
```

Arguments

user - The User object

Return Values

Returns a Person object if the user is found and nil otherwise.

Description

Returns a Person object if the user is found and nil otherwise. Person objects only exist in the current instance of the scene.

Examples

```
local person = User.GetPerson( user )
if person ~= nil then
    friendLockStatus = Person.Lock( person ) -- locking a friend
end
```

See Also

None

User.GetSceneOwner

Brief

Returns the owner of a scene as a User object.

Definition

```
User|nil User.GetSceneOwner()
```

Arguments

None

Return Values

The owner as a User or nil if there is no owner or it hasn't calculated the owner yet.

Description

Returns the owner of a scene as a User object. If called from an apartment it will return the owner of the apartment and if called from a club house it will return the leader of the club house. All other types of scene will return nil. It may initially return nil if you are not the owner of the scene so you should keep calling it each frame until it returns a User. The cases in which this could happen are club houses and when visiting a friend's apartment.

Examples

```
local owner = User.GetSceneOwner()
if owner ~= nil and User.GetName( owner ) == Person.GetName( LocalPlayer.GetPerson() ) then
    print( 'I am the owner' )
end
```

See Also

- User.GetPerson
- Person.GetInstanceSessionMasterId

User.GoToLocation

Brief

Go to the user's location.

Definition

```
User.GoToLocation( User user )
```

Arguments

user - The user object.

Return Values

Description

Go to the user's location.

Examples

```
friends = LocalPlayer.GetFriends()
User.GoToLocation( friends[ 1 ] )
```

See Also

- LocalPlayer.GetFriends

User.InviteToLocation

Brief

Invites the user to the Local player's location.

Definition

```
User.InviteToLocation( User user )
```

Arguments

user - The user object.

Return Values

Description

Invites the user to the Local player's location.

Examples

```
friends = LocalPlayer.GetFriends()
for index, friend in ipairs( friends ) do
    if User.IsInHome( friend ) == true then
        User.InviteToLocation( friend )
    end
end
```

See Also

- LocalPlayer.GetFriends

User.IsFriend

Brief

Returns true if the user is a friend.

Definition

```
boolean User.IsFriend( User user )
```

Arguments

user - The user object.

Return Values

Returns true if the user is a friend and false otherwise.

Description

Returns true if the user is a friend.

Examples

```
friends = LocalPlayer.GetFriends()
for index, friend in ipairs( friends ) do
    if User.IsFriend( friend ) == true then
        print( User.GetName( friend ) .. ' is a friend.' )
    else
        print( 'Something bad has happened.' )
    end
end
```

See Also

- LocalPlayer.GetFriends

User.IsInHome

Brief

Returns true if the user is in PS Home.

Definition

```
boolean User.IsInHome( User user )
```

Arguments

user - The user object.

Return Values

Returns true if the user is in PS Home and false otherwise.

Description

Returns true if the user is in PS Home.

Examples

```
friends = LocalPlayer.GetFriends()
for index, friend in ipairs( friends ) do
    if User.IsInHome( friend ) == true then
        print( User.GetName( friend ) .. ' is in PS Home.' )
    end
end
```

See Also

- LocalPlayer.GetFriends
- User.IsOnline

User.IsOnline

Brief

Returns true if the user is online.

Definition

```
boolean User.IsOnline( User user )
```

Arguments

user - The user object.

Return Values

Returns true if the user is online and false otherwise.

Description

Returns true if the user is online. They are not necessarily in PS Home when online.

Examples

```
friends = LocalPlayer.GetFriends()
for index, friend in ipairs( friends ) do
    if User.IsOnline( friend ) == true then
        print( User.GetName( friend ) .. ' is online.' )
    end
end
```

See Also

- LocalPlayer.GetFriends
- User.IsInHome

Vector4

Vector4.Abs

Brief

Compute the component-wise absolute value of a vector.

Definition

```
Vector4.Abs(Vector4 dest, Vector4 v)
```

Arguments

dest - destination vector to store result.v - vector to negate.

Return Values

None.

Description

Sets dest to [math.abs(v:X()), math.abs(v:Y()), math.abs(v:Z()), math.abs(v:W())].

Examples

```
local v = Vector4.Create(-1, 2, -3)
v:Abs(v)
print(v)        -- [1, 2, 3]
```

See Also

None

Vector4.Add

Brief

Adds one vector or number to another and stores the result in the destination.

Definition

```
Vector4.Add( Vector4 destination, Vector4|number source1, Vector4|number source2 )
```

Arguments

destination - Vector to store the result of the addition.source1 - Vector or number to be added to source2.source2 - Vector or number to be added to source1.

Return Values

None.

Description

Adds one vector or number to another and stores the result in the destination.

Descriptive notation:

destination.x = source1.x|source1 + source2.x|source2

destination.y = source1.y|source1 + source2.y|source2

destination.z = source1.z|source1 + source2.z|source2

destination.w = source1.w|source1 + source2.w|source2

Examples

```
result = Vector4.Create()
v1 = Vector4.Create( 1, 2, 3, 4 )
v2 = Vector4.Create( 4, 3, 2, 1 )
Vector4.Add( result, v1, v2 )
Vector4.Add( result, 3, v2 )
```

See Also

- Vector4.Create

Vector4.Calculate2dLineIntersection

Brief

Calculates a 2-dimensional line intersection.

Definition

```
boolean Vector4.Calculate2dLineIntersection( Vector4 intersectionPoint, Vector4 p0, Vector4 p1,
Vector4 q0, Vector4 q1, boolean infiniteIntersection = false )
```

Arguments

intersectionPoint - The result of the line intersection.p0 - First endpoint of first line.p1 - Second endpoint of first line.q0 - First endpoint of second line.q1 - Second endpoint of second line.infiniteIntersection - true if the supplied lines are to be treated as infinite rays, false otherwise

Return Values

Returns true if lines intersect, false otherwise.

Description

Calculates a 2-dimensional line intersection with the line p, defined by points p0 and p1, with the line q, defined by points q0 and q1. If the infiniteIntersection is true it returns false if p and q are parallel. If infiniteIntersection is false it returns false if the lines are parallel or the intersection does not lie on both p and q.

Examples

```
p0 = Vector4.Create( 1, 0 )
p1 = Vector4.Create( 0, 0 )
q0 = Vector4.Create( 0, 0.5 )
q1 = Vector4.Create( 1, 0.5 )
ip = Vector4.Create()

if Vector4.Calculate2dLineIntersection( ip, p0, p1, q0, q1 ) then
    print( "The lines intersect!" )
end
```

See Also

- Vector4.Create

Vector4.Copy

Brief

Copies the values of one vector into another.

Definition

```
Vector4.Copy( Vector4 destination, Vector4 source )
```

Arguments

destination - The vector to copy to.source - The vector to be copied.

Return Values

None.

Description

Copies the values of the source vector and stores them in the destination. This is used to copy the values as opposed to pointing to the same vector as an assignment would do. Use copy as vec2 = vec1 does not copy values but instead makes vec2 point to vec1.

Examples

```
vec1 = Vector4.Create( 4, 3, 2, 1 )
vec2 = Vector4.Create()
Vector4.Copy( vec2, vec1 )
```

See Also

- Vector4.Create

Vector4.Create

Brief

Creates a new Vector4 object using the specified values.

Definition

```
Vector4 Vector4.Create( Vector4 source )
Vector4 Vector4.Create( number x = 0.0, number y = 0.0, number z = 0.0, number w = 0.0 )
```

Arguments

source - Vector whose values are used to initialize newly created vector.x - Number to be written into the first element of the vector (defaults to 0).y - Number to be written into the second element of the vector (defaults to 0).z - Number to be written into the third element of the vector (defaults to 0).w - Number to be written into the fourth element of the vector (defaults to 0).

Return Values

A new Vector4 instance.

Description

Creates a new Vector4 object using the specified values.

Examples

```
-- Create a default zero (0, 0, 0, 0) vector
vec1 = Vector4.Create()

-- Create a vector with supplied values
vec2 = Vector4.Create( 1, 2, 3, 4 )

-- Create a vector using the values of another
vec3 = Vector4.Create( vec2 )
```

See Also

- Vector4.CreateBroadcast

Vector4.CreateBroadcast

Brief

Creates a new Vector4 and sets all its elements to the specified value.

Definition

```
Vector4 Vector4.CreateBroadcast( number value )
```

Arguments

value - The value to broadcast to all the elements of the new vector.

Return Values

A new Vector4 instance.

Description

Creates a new Vector4 and sets all its elements to the specified value.

Examples

```
vecHalf = Vector4.CreateBroadcast( 0.5 )
vec = Vector4.Create( 1, 2, 3 )
Vector4.Multiply( vec, vec, vecHalf )
```

See Also

- Vector4.Create

Vector4.Cross

Brief

Performs a cross product between two 3-dimensional vectors.

Definition

```
Vector4.Cross( Vector4 destination, Vector4 src1, Vector4 src2 )
```

Arguments

destination - Vector to store the result of the cross product.src1 - First vector to take part in the cross product.src2 - Second vector to take part in the cross product.

Return Values

None.

Description

Performs a 3-dimensional cross product between src1 and src2 storing the result in destination. The vector generated will be perpendicular to both src1 and src2.

Vector algebra notation:

src1 X src2

Descriptive notation:

destination.x = (src1.y * src2.z) - (src1.z * src2.y)

destination.y = (src1.z * src2.x) - (src1.x * src2.z)

destination.z = (src1.x * src2.y) - (src1.y * src2.x)

Examples

```
xAxis = Vector4.Create( 1, 0, 0 )
yAxis = Vector4.Create( 0, 1, 0 )
result = Vector4.Create()
Vector4.Cross( result, xAxis, yAxis )
```

See Also

- Vector4.Create

Vector4.Dist2Sqr

Brief

Computes the squared Euclidean distance between two points in 2D space.

Definition

```
number Vector4.Dist2Sqr( Vector4 p1, Vector4 p2 )
```

Arguments

p1 - A point in 2D space.p2 - A second point in 2D space.

Return Values

The squared Euclidean distance between the two points.

Description

Computes the squared Euclidean distance between two points in 2D space.

Vector algebra notation:

((p2-p1) dot (p2-p1))

Examples

```
v1 = Vector4.Create(1, 2)
v2 = Vector4.Create(2, 3)
print("Distance between", v1, "and", v2, "is:", math.sqrt(Vector4.Dist2Sqr(v1, v2)))
```

See Also

- Vector4.Dist3Sqr
- Vector4.Dist4Sqr

Vector4.Dist3Sqr

Brief

Computes the squared Euclidean distance between two points in 3D space.

Definition

```
number Vector4.Dist3Sqr( Vector4 p1, Vector4 p2 )
```

Arguments

p1 - A point in 3D space.p2 - A second point in 3D space.

Return Values

The squared Euclidean distance between the two points.

Description

Computes the squared Euclidean distance between two points in 3D space.

Vector algebra notation:

((p2-p1) dot (p2-p1))

Examples

```
v1 = Vector4.Create(1, 2, 3)
v2 = Vector4.Create(2, 3, 4)
print("Distance between", v1, "and", v2, "is:", math.sqrt(Vector4.Dist3Sqr(v1, v2)))
```

See Also

- Vector4.Dist2Sqr
- Vector4.Dist4Sqr

Vector4.Dist4Sqr

Brief

Computes the squared Euclidean distance between two points in 4D space.

Definition

```
number Vector4.Dist4Sqr( Vector4 p1, Vector4 p2 )
```

Arguments

p1 - A point in 4D space.p2 - A second point in 4D space.

Return Values

The squared Euclidean distance between the two points.

Description

Computes the squared Euclidean distance between two points in 4D space.

Vector algebra notation:

((p2-p1) dot (p2-p1))

Examples

```
v1 = Vector4.Create( 1, 2, 3, 4 )
v2 = Vector4.Create( 2, 3, 4, 5 )
print("Distance between", v1, "and", v2, "is:", math.sqrt(Vector4.Dist4Sqr(v1, v2)))
```

See Also

- Vector4.Dist2Sqr
- Vector4.Dist3Sqr

Vector4.Divide

Brief

Divides one vector or scalar by the other vector or scalar.

Definition

```
Vector4.Divide( Vector4 destination, Vector4|number source1, Vector4|number source2 )
```

Arguments

destination - Vector to store the result of the division.source1 - The dividend of the vector division.source2 - The divisor of the vector division.

Return Values

None.

Description

Divides source1 by source2 storing the result in destination.

Descriptive notation:

destination.x = source1.x|source1 / source2.x|source2

destination.y = source1.y|source1 / source2.y|source2

destination.z = source1.z|source1 / source2.z|source2

destination.w = source1.w|source1 / source2.w|source2

Examples

```
result = Vector4.Create()
v1 = Vector4.Create( 1, 2, 3, 4 )
v2 = Vector4.Create( 4, 3, 2, 1 )
Vector4.Divide( result, v1, v2 )
Vector4.Divide( result, 1.0, v2 )
```

See Also

- Vector4.Create

Vector4.Dot2

Brief

Performs a dot product between two 2-dimensional vectors.

Definition

```
number Vector4.Dot2( Vector4 src1, Vector4 src2 )
```

Arguments

src1 - First vector to take part in the dot product.src2 - Second vector to take part in the dot product.

Return Values

The number representing the dot product of the two supplied 2-dimensional vectors.

Description

Performs a 2-dimensional dot product with src1 and src2 returning the result.

Vector algebra notation:

src1 . src2

Descriptive notation:

returnvalue = (src1.x * src2.x) + (src1.y * src2.y)

Examples

```
v1 = Vector4.Create( 1, 2 )
v2 = Vector4.Create( 4, 3 )
result = Vector4.Dot2( v1, v2 )
```

See Also

- Vector4.Create
- Vector4.Dot3
- Vector4.Dot4

Vector4.Dot3

Brief

Performs a dot product between two 3-dimensional vectors.

Definition

```
number Vector4.Dot3( Vector4 src1, Vector4 src2 )
```

Arguments

src1 - First vector to take part in the dot product.src2 - Second vector to take part in the dot product.

Return Values

The number representing the dot product of the two supplied 3-dimensional vectors.

Description

Performs a 3-dimensional dot product with src1 and src2 returning the result.

Vector algebra notation:

src1 . src2

Descriptive notation:

returnvalue = (src1.x * src2.x) + (src1.y * src2.y) + (src1.z * src2.z)

Examples

```
v1 = Vector4.Create( 1, 2, 3 )
v2 = Vector4.Create( 4, 3, 2 )
result = Vector4.Dot3( v1, v2 )
```

See Also

- Vector4.Create
- Vector4.Dot2
- Vector4.Dot4

Vector4.Dot4

Brief

Performs a dot product between two 4-dimensional vectors.

Definition

```
number Vector4.Dot4( Vector4 src1, Vector4 src2 )
```

Arguments

src1 - First vector to take part in the dot product.src2 - Second vector to take part in the dot product.

Return Values

The number representing the dot product of the two supplied 4-dimensional vectors.

Description

Performs a 4-dimensional dot product with src1 and src2 returning the result.

Vector algebra notation:

src1 . src2

Descriptive notation:

returnvalue = (src1.x * src2.x) + (src1.y * src2.y) + (src1.z * src2.z) + (src1.w * src2.w)

Examples

```
v1 = Vector4.Create( 1, 2, 3, 4 )
v2 = Vector4.Create( 4, 3, 2, 1 )
result = Vector4.Dot4( v1, v2 )
```

See Also

- Vector4.Create
- Vector4.Dot2
- Vector4.Dot3

Vector4.HorizAdd3

Brief

Compute the horizontal sum of a 3D vector.

Definition

```
number Vector4.HorizAdd3(Vector4 v)
```

Arguments

v - input vector.

Return Values

The sum of the X, Y and Z components of the input vector.

Description

Returns v:X() + v:Y() + v:Z(). W is ignored.

Examples

```
local v = Vector4.Create(1, 2, 3)
print(v:HorizAdd3())        -- 6
```

See Also

- Vector4.HorizAdd4

Vector4.HorizAdd4

Brief

Compute the horizontal sum of a 4D vector.

Definition

```
number Vector4.HorizAdd4(Vector4 v)
```

Arguments

v - input vector.

Return Values

The sum of the X, Y, Z and W components of the input vector.

Description

Returns v:X() + v:Y() + v:Z() + v:W().

Examples

```
local v = Vector4.Create(1, 2, 3, 4)
print(v:HorizAdd4())        -- 10
```

See Also

- Vector4.HorizAdd3

Vector4.Length2

Brief

Calculates the length of a 2-dimensional vector.

Definition

```
number Vector4.Length2( Vector4 vector )
```

Arguments

vector - Source vector for the calculation.

Return Values

A number holding the magnitude of the specified 2-dimensional vector.

Description

Calculates the length of the specified 2-dimensional vector.

Vector algebra notation:

vector	or sqrt(vector dot vector)

Descriptive notation:

returnvalue = sqrt((vector.x * vector.x) + (vector.y * vector.y))

Examples

```
v1 = Vector4.Create( 1, 2 )
print( "Vector magnitude is: " .. Vector4.Length2( v1 ) )
```

See Also

- Vector4.Create
- Vector4.Length3
- Vector4.Length4

Vector4.Length2Sqr

Brief

Calculates the squared length of a 2-dimensional vector.

Definition

```
number Vector4.Length2Sqr( Vector4 vector )
```

Arguments

vector - Source vector for the calculation.

Return Values

A number holding the squared magnitude of the specified 2-dimensional vector.

Description

Calculates the squared length of the specified 2-dimensional vector.

Vector algebra notation:

(vector dot vector)

Descriptive notation:

returnvalue = (vector.x * vector.x) + (vector.y * vector.y)

Examples

```
v1 = Vector4.Create( 1, 2 )
print( "Vector squared magnitude is: " .. Vector4.Length2Sqr( v1 ) )
```

See Also

- Vector4.Create
- Vector4.Length3Sqr
- Vector4.Length4Sqr
- Vector4.Length2

Vector4.Length3

Brief

Calculates the length of a 3-dimensional vector.

Definition

```
number Vector4.Length3( Vector4 vector )
```

Arguments

vector - Source vector for the calculation.

Return Values

A number holding the magnitude of the specified 3-dimensional vector.

Description

Calculates the length of the specified 3-dimensional vector.

Vector algebra notation:

vector	or sqrt(vector dot vector)

Descriptive notation:

returnvalue = sqrt((vector.x * vector.x) + (vector.y * vector.y) + (vector.z * vector.z))

Examples

```
v1 = Vector4.Create( 1, 2, 3 )
print( "Vector magnitude is: " .. Vector4.Length3( v1 ) )
```

See Also

- Vector4.Create
- Vector4.Length2
- Vector4.Length4

Vector4.Length3Sqr

Brief

Calculates the squared length of a 3-dimensional vector.

Definition

```
number Vector4.Length3Sqr( Vector4 vector )
```

Arguments

vector - Source vector for the calculation.

Return Values

A number holding the squared magnitude of the specified 3-dimensional vector.

Description

Calculates the squared length of the specified 3-dimensional vector.

Vector algebra notation:

(vector dot vector)

Descriptive notation:

returnvalue = (vector.x * vector.x) + (vector.y * vector.y) + (vector.z * vector.z)

Examples

```
v1 = Vector4.Create( 1, 2, 3 )
print( "Vector squared magnitude is: " .. Vector4.Length34Sqr( v1 ) )
```

See Also

- Vector4.Create
- Vector4.Length2Sqr
- Vector4.Length4Sqr
- Vector4.Length3

Vector4.Length4

Brief

Calculates the length of a 4-dimensional vector.

Definition

```
number Vector4.Length4( Vector4 vector )
```

Arguments

vector - Source vector for the calculation.

Return Values

A number holding the magnitude of the specified 4-dimensional vector.

Description

Calculates the length of the specified 4-dimensional vector.

Vector algebra notation:

vector	or sqrt(vector dot vector)

Descriptive notation:

returnvalue = sqrt((vector.x * vector.x) + (vector.y * vector.y) + (vector.z * vector.z) + (vector.w * vector.w))

Examples

```
v1 = Vector4.Create( 1, 2, 3, 4 )
print( "Vector magnitude is: " .. Vector4.Length4( v1 ) )
```

See Also

- Vector4.Create
- Vector4.Length2
- Vector4.Length3

Vector4.Length4Sqr

Brief

Calculates the squared length of a 4-dimensional vector.

Definition

```
number Vector4.Length4Sqr( Vector4 vector )
```

Arguments

vector - Source vector for the calculation.

Return Values

A number holding the squared magnitude of the specified 4-dimensional vector.

Description

Calculates the squared length of the specified 4-dimensional vector.

Vector algebra notation:

(vector dot vector)

Descriptive notation:

returnvalue = (vector.x * vector.x) + (vector.y * vector.y) + (vector.z * vector.z) + (vector.w * vector.w)

Examples

```
v1 = Vector4.Create( 1, 2, 3, 4 )
print( "Vector squared magnitude is: " .. Vector4.Length4Sqr( v1 ) )
```

See Also

- Vector4.Create
- Vector4.Length2Sqr
- Vector4.Length3Sqr
- Vector4.Length4

Vector4.Max

Brief

Compute the maximum element values of two vectors.

Definition

```
Vector4.Max(Vector4 dest, Vector4 source1, Vector4 source2)
```

Arguments

dest - Vector to hold result. May be the same vector as either of source vectors if desired.source1 - First source vector.source2 - Second source vector.

Return Values

None.

Description

Compares each element in source1 with equivalent element in source2 and stores the greater value in the equivalent element of dest. That is: dest =

Unknown macro: { max(source1.x, source2.x), max(source1.y, source2.y), max(source1.z, source2.z), max(source1.w, source2.w) }

Examples

```
-- assuming verts is an array of Vector4 values
local min = Vector4.Create(Const.MaxNum, Const.MaxNum, Const.MaxNum)
local max = Vector4.Create(-Const.MaxNum, -Const.MaxNum, -Const.MaxNum)
for i = 1, numVerts do
    Vector4.Min(min, min, verts[i])
    Vector4.Max(max, max, verts[i])
end
-- min and max now store bounds of vert list
```

See Also

- Vector4.Min
- StreamOps.BoundsVec4

Vector4.Min

Brief

Compute the minimum element values of two vectors.

Definition

```
Vector4.Min(Vector4 dest, Vector4 source1, Vector4 source2)
```

Arguments

dest - Vector to hold result. May be the same vector as either of source vectors if desired.source1 - First source vector.source2 - Second source vector.

Return Values

None.

Description

Compares each element in source1 with equivalent element in source2 and stores the lower value in the equivalent element of dest. That is: dest =

Unknown macro: { min(source1.x, source2.x), min(source1.y, source2.y), min(source1.z, source2.z), min(source1.w, source2.w) }

Examples

```
-- assuming verts is an array of Vector4 values
local min = Vector4.Create(Const.MaxNum, Const.MaxNum, Const.MaxNum)
local max = Vector4.Create(-Const.MaxNum, -Const.MaxNum, -Const.MaxNum)
for i = 1, numVerts do
    Vector4.Min(min, min, verts[i])
    Vector4.Max(max, max, verts[i])
end
-- min and max now store bounds of vert list
```

See Also

- Vector4.Max
- StreamOps.BoundsVec4

Vector4.Multiply

Brief

Multiplies two vectors or scalars together storing the result as a vector.

Definition

```
Vector4.Multiply( Vector4 destination, Vector4|number source1, Vector4|number source2 )
```

Arguments

destination - Vector to store the result of the multiplication.source1 - The first argument for the multiplication, can be a Vector4 or a scalar.source2 - The second argument for the multiplication, can be a Vector4 or a scalar.

Return Values

None.

Description

Multiplies source1 and source2 together storing the result in the destination vector.

Descriptive notation:

destination.x = source1.x|source1 * source2.x|source2

destination.y = source1.y|source1 * source2.y|source2

destination.z = source1.z|source1 * source2.z|source2

destination.w = source1.w|source1 * source2.w|source2

Examples

```
result = Vector4.Create()
v1 = Vector4.Create( 1, 2, 3, 4 )
v2 = Vector4.Create( 4, 3, 2, 1 )
Vector4.Multiply( result, v1, v2 )
Vector4.Multiply( result, v1, 4.0 )
Vector4.Multiply( result, -3.0, v2 )
Vector4.Multiply( result, -3.0, 4 )
```

See Also

- Vector4.Create

Vector4.MultiplyAdd

Brief

Multiplies two vectors together and then adds the third vector.

Definition

```
Vector4.MultiplyAdd( Vector4 destination, Vector4|number source1, Vector4|number source2,
Vector4|number source3 )
```

Arguments

destination - Vector to store the result of the multiplication.source1 - The first vector argument for the multiplication.source2 - The second vector argument for the multiplication.source3 - The third vector argument for the addition.

Return Values

None.

Description

Multiplies source1 and source2 together and then adds the third vector, storing the result in destination.

Descriptive notation:

destination.x = (source1.x|source1 * source2.x|source2) + source3.x|source3

destination.y = (source1.y|source1 * source2.y|source2) + source3.y|source3

destination.z = (source1.z|source1 * source2.z|source2) + source3.z|source3

destination.w = (source1.w|source1 * source2.w|source2) + source3.w|source3

Examples

```
result = Vector4.Create()
v1 = Vector4.Create( 1, 2, 3, 4 )
v2 = Vector4.Create( 4, 3, 2, 1 )
v3 = Vector4.Create( 1, 0, 0, 1 )
Vector4.MultiplyAdd( result, v1, v2, v3 )
Vector4.MultiplyAdd( result, 1, 2, 6 )
```

See Also

- Vector4.Create

Vector4.Negate

Brief

Negate a vector in-place.

Definition

```
Vector4.Negate(Vector4 v)
```

Arguments

v - vector to negate.

Return Values

None.

Description

Sets v to -v.

Examples

```
local v = Vector4.Create(1, 2, 3)
v:Negate()
print(v)        -- [-1, -2, -3]
```

See Also

None

Vector4.Normal2

Brief

Returns the normalization of a 2-dimensional vector.

Definition

```
Vector4 Vector4.Normal2( Vector4 vector )
```

Arguments

vector - The vector to normalize.

Return Values

A new Vector4 instance containing a normalized copy of the source vector.

Description

Returns the normalization of the specified 2-dimensional vector (i.e. a new vector of unit length). The source vector is not modified.

Descriptive notation:

length = sqrt(src.x * src.x + src.y * src.y)

returnvec.x = src.x / length

returnvec.y = src.y / length

returnvec.z = 0

returnvec.w = 0

Examples

```
vec = Vector4.Create( 1, 2 )
result = Vector4.Normal2( vec )
print( result )
```

See Also

- Vector4.Create
- Vector4.Normal3
- Vector4.Normal4

Vector4.Normal3

Brief

Returns the normalization of a 3-dimensional vector.

Definition

```
Vector4 Vector4.Normal3( Vector4 src )
```

Arguments

src - The vector to use as a source to normalize.

Return Values

A new Vector4 instance containing a normalized copy of the source vector.

Description

Returns the normalization of the specified 3-dimensional vector (i.e. a new vector of unit length). The source vector is not modified.

Descriptive notation:

length = sqrt(src.x*src.x + src.y*src.y + src.z*src.z)

returnvec.x = src.x / length

returnvec.y = src.y / length

returnvec.z = src.z / length

returnvec.w = 0

Examples

```
vec = Vector4.Create( 1, 2, 3 )
result = Vector4.Normal3( vec )
print( result )
```

See Also

- Vector4.Create
- Vector4.Normal2
- Vector4.Normal4

Vector4.Normal4

Brief

Returns the normalization of a 4-dimensional vector.

Definition

```
Vector4 Vector4.Normal4( Vector4 src )
```

Arguments

src - The vector to use as a source to normalize.

Return Values

A new Vector4 instance containing a normalized copy of the source vector.

Description

Returns the normalization of the specified 4-dimensional vector (i.e. a new vector of unit length). The source vector is not modified.

Descriptive notation:

length = sqrt(src.x*src.x + src.y*src.y + src.z*src.z + src.w*src.w)

returnvec.x = src.x / length

returnvec.y = src.y / length

returnvec.z = src.z / length

returnvec.w = src.z / length

Examples

```
vec = Vector4.Create( 1, 2, 3, 4 )
result = Vector4.Normal4( vec )
print( result )
```

See Also

- Vector4.Create
- Vector4.Normal2
- Vector4.Normal3

Vector4.Normalize2

Brief

Normalizes a vector using Length2 of that vector.

Definition

```
Vector4.Normalize2( Vector4 vector )
```

Arguments

vector - The vector to normalize.

Return Values

None.

Description

Normalizes the specified vector using Length2 of the vector.

Descriptive notation:

length = sqrt(vector.x*vector.x + vector.y*vector.y)

vector.x = vector.x / length

vector.y = vector.y / length

vector.z = 0

vector.w = 0

Examples

```
vector = Vector4.Create( 1, 3, 2, 4 )
Vector4.Normalize2( vector )
print( vector )
```

See Also

- Vector4.Create
- Vector4.Normalize3
- Vector4.Normalize4

Vector4.Normalize3

Brief

Normalizes a vector using Length3 of that vector.

Definition

```
Vector4.Normalize3( Vector4 vector )
```

Arguments

vector - The vector to normalize.

Return Values

None.

Description

Normalizes the specified vector using Length3 of the vector.

Descriptive notation:

length = sqrt(vector.x*vector.x + vector.y*vector.y + vector.z*vector.z)

vector.x = vector.x / length

vector.y = vector.y / length

vector.z = vector.z / length

vector.w = 0

Examples

```
vector = Vector4.Create( 1, 3, 2, 4 )
Vector4.Normalize3( vector )
print( vector )
```

See Also

- Vector4.Create
- Vector4.Normalize2
- Vector4.Normalize4

Vector4.Normalize4

Brief

Normalizes a vector using Length4 of that vector.

Definition

```
Vector4.Normalize4( Vector4 vector )
```

Arguments

vector - The vector to normalize.

Return Values

None.

Description

Normalizes the specified vector using Length4 of the vector.

Descriptive notation:

length = sqrt(vector.x*vector.x + vector.y*vector.y + vector.z*vector.z + vector.w*vector.w)

vector.x = vector.x / length

vector.y = vector.y / length

vector.z = vector.z / length

vector.w = vector.w / length

Examples

```
vector = Vector4.Create( 1, 3, 2, 4 )
Vector4.Normalize4( vector )
print( vector )
```

See Also

- Vector4.Create
- Vector4.Normalize2
- Vector4.Normalize3

Vector4.Recip

Brief

Compute the component-wise reciprocal of a vector.

Definition

```
Vector4.Recip(Vector4 dest, Vector4 v)
```

Arguments

dest - destination vector to store result.v - input vector.

Return Values

None.

Description

Sets dest to [1 / v:X(), 1 / v:Y(), 1 / v:Z(), 1 / v:W()]. If any elements of v are 0 the result is undefined. Note that due to floating point arithmetic precision, Recip(Recip) is not necessarily x.

Examples

```
local v = Vector4.Create(-1, 2, -3)
v:Recip(v)
print(v)        -- [-1, 0.5, -0.333333]
```

See Also

- Vector4.RecipSqrt

Vector4.RecipSqrt

Brief

Compute the component-wise reciprocal square root of a vector.

Definition

```
Vector4.RecipSqrt(Vector4 dest, Vector4 v)
```

Arguments

dest - destination vector to store result.v - input vector.

Return Values

None.

Description

Sets dest to [1 / math.sqrt(v:X()), 1 / math.sqrt(v:Y()), 1 / math.sqrt(v:Z()), 1 / math.sqrt(v:W()]. If any elements of v are <= 0 the result is undefined. This operation is faster than the equivalent combination of Sqrt and Recip.

Examples

```
local v = Vector4.Create(1, 4, 9)
v:RecipSqrt(v)
print(v)        -- [1, 0.5, 0.333333]
```

See Also

- Vector4.Recip
- Vector4.Sqrt

Vector4.Rotate

Brief

Rotate a vector about and axis.

Definition

```
Vector4.Rotate( Vector4 destination, Vector4 source, Vector4 axis, number angle )
```

Arguments

destination - Vector to store the result of the rotation.source - Vector to be rotated.axis - Axis to rotate about.angle - Angle (in degrees) to rotate by.

Return Values

Description

Rotate a vector about and axis.

Examples

```
src = Vector4.Create( 12, 2, -3 )
axis = Vector4.Create( 0, 0.707, -0.707 )
dst = Vector4.Create()
Vector4.Rotate( dst, src, axis, 45 )
```

See Also

- Vector4.Create

Vector4.SetBroadcast

Brief

Sets the all of the vector's elements to the specified value.

Definition

```
Vector4 Vector4.SetBroadcast( Vector4 vector, number value )
```

Arguments

vector - Vector to be set.value - The value to broadcast to all the elements of the vector.

Return Values

None.

Description

Sets the all of the vector's elements to the specified value.

Examples

```
vecHalf = Vector4.Create()
Vector4.SetBroadcast( vecHalf, 0.5 )

vec = Vector4.Create( 1, 2, 3 )
Vector4.Multiply( vec, vec, vecHalf )
```

See Also

- Vector4.Create
- Vector4.CreateBroadcast

Vector4.SetW

Brief

Sets the w value of the vector.

Definition

```
Vector4.SetW( Vector4 vector, number value )
```

Arguments

vector - Vector to be set.value - A numerical value that is copied into the w component of the vector.

Return Values

None.

Description

Sets the w value of the vector.

Examples

```
vec = Vector4.Create( 1, 2, 3, 4 )
Vector4.SetW( vec, 6 )
```

See Also

- Vector4.Create
- Vector4.SetX
- Vector4.SetY
- Vector4.SetZ

Vector4.SetX

Brief

Sets the x value of the vector.

Definition

```
Vector4.SetX( Vector4 vector, number value )
```

Arguments

vector - Vector to be set.value - A numerical value that is copied into the x component of the vector.

Return Values

None.

Description

Sets the x value of the vector.

Examples

```
vec = Vector4.Create( 1, 2, 3, 4 )
Vector4.SetX( vec, 6 )
```

See Also

- Vector4.Create
- Vector4.SetY
- Vector4.SetZ
- Vector4.SetW

Vector4.SetXyzw

Brief

Sets a Vector4 object with the specified values.

Definition

```
Vector4.SetXyzw( Vector4 vector, number x = 0, number y = 0, number z = 0, number w = 0 )
```

Arguments

vector - Vector to be set.x - Number to be written into the first element of the vector (defaults to 0).y - Number to be written into the second element of the vector (defaults to 0).z - Number to be written into the third element of the vector (defaults to 0).w - Number to be written into the fourth element of the vector (defaults to 0).

Return Values

None.

Description

Sets a Vector4 object with the specified values.

Examples

```
vec = Vector4.Create()
Vector4.SetXyzw( vec, 1, 2, 3, 4 )
```

See Also

- Vector4.Create

Vector4.SetY

Brief

Sets the y value of the vector.

Definition

```
Vector4.SetY( Vector4 vector, number value )
```

Arguments

vector - Vector to be set.value - A numerical value that is copied into the y component of the vector.

Return Values

None.

Description

Sets the y value of the vector.

Examples

```
vec = Vector4.Create( 1, 2, 3, 4 )
Vector4.SetY( vec, 6 )
```

See Also

- Vector4.Create
- Vector4.SetX
- Vector4.SetZ
- Vector4.SetW

Vector4.SetZ

Brief

Sets the z value of the vector.

Definition

```
Vector4.SetZ( Vector4 vector, number value )
```

Arguments

vector - Vector to be set.value - A numerical value that is copied into the z component of the vector.

Return Values

None.

Description

Sets the z value of the vector.

Examples

```
vec = Vector4.Create( 1, 2, 3, 4 )
Vector4.SetZ( vec, 6 )
```

See Also

- Vector4.Create
- Vector4.SetX
- Vector4.SetY
- Vector4.SetW

Vector4.Sign

Brief

Compute the sign (-1 or 1) of all elements in a vector.

Definition

```
Vector4.Sign(Vector4 dest, Vector4 v)
```

Arguments

dest - destination vector to store result.v - input vector.

Return Values

None.

Description

Each element of dest is set to the sign (-1 or 1) of the equivalent element of v. 0 is considered positive and thus becomes 1, if a separate 0 value is required use the Signum function.

Examples

```
local v = Vector4.Create(-3.1, 0, -0.4, 1.2)
v:Sign(v)
print(v)        -- [-1, 1, -1, 1]
```

See Also

- Vector4.Signum

Vector4.Signum

Brief

Compute the sign (-1, 0 or 1) of all elements in a vector.

Definition

```
Vector4.Signum(Vector4 dest, Vector4 v)
```

Arguments

dest - destination vector to store result.v - input vector.

Return Values

None.

Description

Each element of dest is set to the sign (-1, 0 or 1) of the equivalent element of v. If a separate 0 value is not required, use the Sign function.

Examples

```
local v = Vector4.Create(-3.1, 0, -0.4, 1.2)
v:Signum(v)
print(v)        -- [-1, 0, -1, 1]
```

See Also

- Vector4.Sign

Vector4.Sqrt

Brief

Compute the component-wise square root of a vector.

Definition

```
Vector4.Sqrt(Vector4 dest, Vector4 v)
```

Arguments

dest - destination vector to store result.v - input vector.

Return Values

None.

Description

Sets dest to [math.sqrt(v:X()), math.sqrt(v:Y()), math.sqrt(v:Z()), math.sqrt(v:W())]. If any elements of v are < 0 the result is undefined.

Examples

```
local v = Vector4.Create(1, 4, 9)
v:Sqrt(v)
print(v)        -- [1, 2, 3]
```

See Also

- Vector4.RecipSqrt

Vector4.Subtract

Brief

Subtracts one vector from another.

Definition

```
Vector4.Subtract( Vector4 destination, Vector4|number source1, Vector4|number source2 )
```

Arguments

destination - Vector to store the result of the subtraction.source1 - The minuend for the subtraction.source2 - The subtrahend for the subtraction.

Return Values

None.

Description

Subtracts source2 from source1 storing the result in destination.

Descriptive notation:

destination.x = source1.x|source1 - source2.x|source2

destination.y = source1.y|source1 - source2.y|source2

destination.z = source1.z|source1 - source2.z|source2

destination.w = source1.w|source1 - source2.w|source2

Examples

```
result = Vector4.Create()
v1 = Vector4.Create( 1, 2, 3, 4 )
v2 = Vector4.Create( 4, 3, 2, 1 )
Vector4.Subtract( result, v1, v2 )
Vector4.Subtract( result, v1, 3 )
```

See Also

- Vector4.Create

Vector4.Transform

Brief

Transforms a vector by a matrix.

Definition

```
Vector4.Transform( Vector4 destination, Matrix44 matrix, Vector4 source )
Vector4.Transform( table destination, Matrix44 matrix, table source )
```

Arguments

destination - Vector or table of vectors to store the result of the multiplication.matrix - Matrix to transform the source vector or table of vectors.source - Vector or table of vectors to be transformed.

Return Values

Description

Transforms the specified source vector by a matrix and stores the resultant vector in the destination. It is equivalent to:

destination = matrix * source
It is also possible to transform a table of vectors by the specified matrix. The result is stored in a destination table with the same number of vectors as the source table. This can be considerably faster than transforming vectors individually.

Examples

```
-- Create the matrix for the example code below
matrix = Matrix44.Create()
Matrix44.SetRotationXyz( matrix, 45, 12, 7 )

-- Example 1: Transform a single vector by a matrix
src = Vector4.Create( 12, 2, -3, 1 )
dst = Vector4.Create()
Vector4.Transform( dst, matrix, src )

-- Example 2: Transform a table of vectors by a matrix
source = {}
destination = {}
for v = 1, 10 do
    source[v] = Vector4.Create(v, v, v, v)
    destination[v] = Vector4.Create()
end
Vector4.Transform( destination, matrix, source )
```

See Also

- Matrix44.Create
- Vector4.Create

Vector4.TransformPoint

Brief

Transforms a point by a matrix.

Definition

```
Vector4.TransformPoint( Vector4 destination, Matrix44 matrix, Vector4 source )
Vector4.TransformPoint( table destination, Matrix44 matrix, table source )
```

Arguments

destination - Point or table of points to store the result of the multiplication.matrix - Matrix to transform the source vector or table of vectors.source - Point or table of points to be transformed.

Return Values

Description

Transforms the specified source point by a matrix and stores the resultant point in the destination. It is equivalent to:

destination = matrix * source (where source.w is substituted with 1.0)

It is also possible to transform a table of points by the specified matrix. The result is stored in a destination table with the same number of points as the source table. This can be considerably faster than transforming points individually.

Examples

```
-- Create the matrix for the example code below
matrix = Matrix44.Create()
Matrix44.SetRotationXyz( matrix, 45, 12, 7 )

-- Example 1: Transform a single point by a matrix
src = Vector4.Create( 12, 2, -3 )
dst = Vector4.Create()
Vector4.TransformPoint( dst, matrix, src )

-- Example 2: Transform a table of points by a matrix
source = {}
destination = {}
for v = 1, 10 do
    source[v] = Vector4.Create(v, v, v)
    destination[v] = Vector4.Create()
end
Vector4.TransformPoint( destination, matrix, source )
```

See Also

- Matrix44.Create
- Vector4.Create

Vector4.TransformVector

Brief

Transforms a vector by a matrix.

Definition

```
Vector4.TransformVector( Vector4 destination, Matrix44 matrix, Vector4 source )
Vector4.TransformVector( table destination, Matrix44 matrix, table source )
```

Arguments

destination - Vector or table of vectors to store the result of the multiplication.matrix - Matrix to transform the source vector or table of vectors.source - Vector or table of vectors to be transformed.

Return Values

Description

Transforms the specified source vector by a matrix and stores the resultant vector in the destination. It is equivalent to:

destination = matrix * source (where source.w is substituted with 0.0)

It is also possible to transform a table of vectors by the specified matrix. The result is stored in a destination table with the same number of vectors as the source table. This can be considerably faster than transforming vectors individually.

Examples

```
-- Create the matrix for the example code below
matrix = Matrix44.Create()
Matrix44.SetRotationXyz( matrix, 45, 12, 7 )

-- Example 1: Transform a single vector by a matrix
src = Vector4.Create( 12, 2, -3 )
dst = Vector4.Create()
Vector4.TransformVector( dst, matrix, src )

-- Example 2: Transform a table of vectors by a matrix
source = {}
destination = {}
for v = 1, 10 do
    source[v] = Vector4.Create(v, v, v)
    destination[v] = Vector4.Create()
end
Vector4.TransformVector( destination, matrix, source )
```

See Also

- Matrix44.Create
- Vector4.Create

Vector4.W

Brief

Returns the w value of the vector.

Definition

```
number Vector4.W( Vector4 vector )
```

Arguments

vector - Vector to be queried.

Return Values

Returns the fourth element of the vector, the w component.

Description

Returns the w value of the vector.

Examples

```
vec = Vector4.Create( 1, 2, 3, 4 )
print( "Value of W is: " .. Vector4.W( vec ) )
```

See Also

- Vector4.Create
- Vector4.X
- Vector4.Y

- Vector4.Z

Vector4.X

Brief

Returns the x value of the vector.

Definition

```
number Vector4.X( Vector4 vector )
```

Arguments

vector - Vector to be queried.

Return Values

The first element of the vector, the x component.

Description

Returns the x value of the vector.

Examples

```
vec = Vector4.Create( 10 )
print( "Value of X is: " .. Vector4.X( vec ) )
```

See Also

- Vector4.Create
- Vector4.Y
- Vector4.Z
- Vector4.W

Vector4.Y

Brief

Returns the y value of the vector.

Definition

```
number Vector4.Y( Vector4 vector )
```

Arguments

vector - Vector to be queried.

Return Values

Returns the second element of the vector, the y component.

Description

Returns the y value of the vector.

Examples

```
vec = Vector4.Create( 1, 2 )
print( "Value of Y is: " .. Vector4.Y( vec ) )
```

See Also

- Vector4.Create
- Vector4.X
- Vector4.Z
- Vector4.W

Vector4.Z

Brief

Returns the z value of the vector.

Definition

```
number Vector4.Z( Vector4 vector )
```

Arguments

vector - Vector to be queried.

Return Values

Returns the third element of the vector, the z component.

Description

Returns the z value of the vector.

Examples

```
vec = Vector4.Create( 1, 2, 3 )
print( "Value of Z is: " .. Vector4.Z( vec ) )
```

See Also

- Vector4.Create
- Vector4.X
- Vector4.Y
- Vector4.W

View

View.CalculatePickRay

Brief

Calculate a pick ray through a specified screen point into the scene.

Definition

```
boolean View.CalculatePickRay(Vector4 screenPoint, Vector4 rayOrigin, Vector4 rayDirection)
boolean View.CalculatePickRay(number x, number y, Vector4 rayOrigin, Vector4 rayDirection)
```

Arguments

screenPoint - Screen point to calculate pick ray for.rayOrigin - Vector4 to return ray origin into.rayDirection - Vector4 to return ray direction vector into.x - Screen point X coordinate, in the range [0 (left) .. 1280 (right)].y - Screen point Y coordinate, in the range [0 (top) .. 720 (bottom)].

Return Values

True on success, false on failure (only in case of no valid camera view active).

Description

This function will calculate the origin and direction of a ray through the specified screen-space point into the scene, based on the camera properties. It is effectively the inverse operation to View.WorldToScreen, except that the ray indicates all the (infinite) points that map onto the specified screen-space point. The ray origin will be coplanar with the near clip plane, the ray direction is not normalized but is rather scaled to the distance between the near and far clip planes.

Examples

```
RETICLE_OFFSET_X = 0
RETICLE_OFFSET_Y = 10
local pickOrigin = Vector4.Create()
local pickDir = Vector4.Create()
if (View.CalculatePickRay(RETICLE_OFFSET_X, RETICLE_OFFSET_Y, pickOrigin, pickDir)) then
    g_raycast = Collision.CreateRaycast()
    g_raycast:Start(pickOrigin, pickOrigin + pickDir)
    while (not g_raycast:IsReady()) do
        coroutine.yield()
    end
    if (g_raycast:HasHit()) then
        local entity = g_raycast:GetHitEntity()
        if (entity and entity:GetUniqueId() == targetEntityId) then
            print("You hit the target!")
            score = score + 10
        end
    end
end
```

See Also

- View.WorldToScreen
- Collision.CreateRaycast
- Raycast.Start

View.WorldToScreen

Brief

Converts a 3d world position to a screen position.

Definition

```
boolean View.WorldToScreen( Vector4 screenPosition, Vector4 worldPosition )
```

Arguments

screenPosition - The 2D screen position to write the result to. The screen space coordinates are in
the range [0,0] (top left) -> [1280,720] (bottom right). Z is set to 0, and W is set to 1.worldPosition - The 3d position in the world.

Return Values

Returns true if it could calculate it, even if it is off screen.

Description

Converts a 3d world position to a 2d screen position if possible. It can return screen coordinates that are actually off screen as you may want to render text for instance that is partially visible.

Examples

```
if View.WorldToScreen( screenPos, worldPos ) == true then
    RenderMyText( screenPos, 'hello' )
end
```

See Also

None

Wardrobe

Wardrobe.ResetCommercePoints

Brief

Reset all commerce point URLs back to default values.

Definition

```
Wardrobe.ResetCommercePoints()
```

Arguments

None

Return Values

None.

Description

Reset all commerce point URLs back to default values.

Examples

```
Wardrobe.ResetCommercePoints()
```

See Also

- Wardrobe.SetCommercePoint

Wardrobe.SetCommercePoint

Brief

Set the commerce point URL for a specific gender/clothing-category combination.

Definition

```
Wardrobe.SetCommercePoint( enum genderId, enum categoryId, string url )
```

Arguments

genderId - A WardrobeGender enum.categoryId - A WardrobeCategory enum.url - The URL of the commerce point definition file.

Return Values

None.

Description

Set the commerce point URL for a specific gender/clothing-category combination. Available gender ids are:
WardrobeGender.Male
WardrobeGender.Female Available category ids are:
WardrobeCategory.All
WardrobeCategory.Hats
WardrobeCategory.Torso
WardrobeCategory.Legs
WardrobeCategory.Hands
WardrobeCategory.Feet
WardrobeCategory.Jewellery
WardrobeCategory.Glasses
WardrobeCategory.Headphones
WardrobeCategory.Outfits If the URL starts with "http:", "https:", etc, it will be considered absolute; otherwise, it will be considered relative to the content server CommercePoints directory.

Examples

```
Wardrobe.SetCommercePoint( WardrobeGender.Male, WardrobeCategory.Torso,
"http://absolute.path.to/commercepoint.xml" )
Wardrobe.SetCommercePoint( WardrobeGender.Female, WardrobeCategory.Hands,
"relative/path/to/commercepoint.xml" )
```

See Also

- Wardrobe.ResetCommercePoints

Xml

Xml.Create

Brief

Creates a new instance of an XML parser.

Definition

```
Xml Xml.Create()
```

Arguments

None

Return Values

A new instance of an XML parser.

Description

Creates a new instance of an XML parser.

Examples

```
local parser = Xml.Create()
```

See Also

- Xml.SetData

Xml.FindAttribute

Brief

Finds a named attribute, or iterates through the available attributes.

Definition

```
boolean Xml.FindAttribute( Xml parser, string attributeName )
```

Arguments

parser - The instance of the XML parser.attributeName - The attribute to search for, or nil to iterate all attributes.

Return Values

True if the attribute was found. If iterating, returns true if there are further attributes or
false if the iteration is complete.

Description

If a valid name is specified, finds the named attribute of the current element. If nil is passed as the name, the function will iterate the attributes in an implementation-defined order with each call. The iteration is complete when the function returns false.

Examples

```
local found = Xml.FindAttribute( parser, "type" )
while ( parser2:FindAttribute( nil ) ) do
    print( parser2:GetAttributeValueText() )
end
```

See Also

None

Xml.FindElement

Brief

Finds the first element that matches the named element.

Definition

```
boolean Xml.FindElement( Xml parser, string elementName )
```

Arguments

parser - The instance of the XML parser.elementName - The name of the XML element to search for. Can be nil.

Return Values

true if the element was found.

Description

Finds the first element at the current level that matches the named element. If nil is passed in as the element name, then the first element is found.

Examples

```
local found = Xml.FindElement( parser, "xml" )
```

See Also

- Xml.Create
- Xml.FindNextElement

Xml.FindNextElement

Brief

Finds the next element that matches the named element.

Definition

```
boolean Xml.FindNextElement( Xml parser, string elementName )
```

Arguments

parser - The instance of the XML parser.elementName - The name of the XML element to search for. Can be nil.

Return Values

true if the element was found.

Description

Finds the next element at the current level that matches the named element. If nil is passed in, then the next element regardless of its name is found.

Examples

```
local found = Xml.FindNextElement( parser, nil )
```

See Also

- Xml.Create
- Xml.FindElement

Xml.GetAttributeName

Brief

Returns the current attribute name, or nil if there is no current attribute.

Definition

```
string Xml.GetAttributeName( Xml parser )
```

Arguments

parser - The instance of the XML parser.

Return Values

The attribute name as a string, or nil if there is no current attribute.

Description

Returns the current attribute name. This will match the last name specified to FindAttribute, or if FindAttribute is used to iterate the attributes, the current attribute name.

Examples

```
while ( parser:FindAttribute( nil ) ) do
    print( parser:GetAttributeName(), "=", parser:GetAttributeValueText() )
end
```

See Also

- Xml.FindAttribute

Xml.GetAttributeValueBool

Brief

Returns the value of the current attribute.

Definition

```
boolean Xml.GetAttributeValueBool( Xml parser )
```

Arguments

parser - The instance of the XML parser.

Return Values

The attribute value as a boolean.

Description

Returns the value of the current attribute.

Examples

```
local value = Xml.GetAttributeValueBool( parser )
```

See Also

None

Xml.GetAttributeValueFloat

Brief

Returns the value of the current attribute.

Definition

```
number Xml.GetAttributeValueFloat( Xml parser )
```

Arguments

parser - The instance of the XML parser.

Return Values

The attribute value as a float.

Description

Returns the value of the current attribute.

Examples

```
local value = Xml.GetAttributeValueFloat( parser )
```

See Also

None

Xml.GetAttributeValueInt

Brief

Returns the value of the current attribute.

Definition

```
number Xml.GetAttributeValueInt( Xml parser )
```

Arguments

parser - The instance of the XML parser.

Return Values

The attribute value as an integer number.

Description

Returns the value of the current attribute.

Examples

```
local value = Xml.GetAttributeValueInt( parser )
```

See Also

None

Xml.GetAttributeValueText

Brief

Returns the value of the current attribute.

Definition

```
string Xml.GetAttributeValueText( Xml parser )
```

Arguments

parser - The instance of the XML parser.

Return Values

The attribute value as a string.

Description

Returns the value of the current attribute.

Examples

```
local value = Xml.GetAttributeValueText( parser )
```

See Also

None

Xml.GetElementName

Brief

Returns the name of the current XML element.

Definition

```
string Xml.GetElementName( Xml parser )
```

Arguments

parser - The instance of the XML parser.

Return Values

The name of the current XML element.

Description

Returns the name of the current XML elemet.

Examples

```
local name = Xml.GetElementName( parser )
```

See Also

None

Xml.GetElementValueBool

Brief

Returns the value of the current XML element.

Definition

```
boolean Xml.GetElementValueBool( Xml parser )
```

Arguments

parser - The instance of the XML parser.

Return Values

The element value as a boolean.

Description

Returns the value of the current XML element.

Examples

```
local value = Xml.GetElementValueBool( parser )
```

See Also

None

Xml.GetElementValueFloat

Brief

Returns the value of the current XML element.

Definition

```
number Xml.GetElementValueFloat( Xml parser )
```

Arguments

parser - The instance of the XML parser.

Return Values

The element value as a float.

Description

Returns the value of the current XML element.

Examples

```
local value = Xml.GetElementValueFloat( parser )
```

See Also

None

Xml.GetElementValueInt

Brief

Returns the value of the current XML element.

Definition

```
number Xml.GetElementValueInt( Xml parser )
```

Arguments

parser - The instance of the XML parser.

Return Values

The element value as an int.

Description

Returns the value of the current XML element.

Examples

```
local value = Xml.GetElementValueInt( parser )
```

See Also

None

Xml.GetElementValueText

Brief

Returns the value of the current XML element.

Definition

```
string Xml.GetElementValueText( Xml parser )
```

Arguments

parser - The instance of the XML parser.

Return Values

The element value as a string.

Description

Returns the value of the current XML element.

Examples

```
local value = Xml.GetElementValueText( parser )
```

See Also

None

Xml.IntoElement

Brief

Navigate into an XML element.

Definition

```
boolean Xml.IntoElement( Xml parser )
```

Arguments

parser - The instance of the XML parser.

Return Values

true if it was possible to navigate into the element.

Description

Navigate into an XML element so that its childen can be examined.

Examples

```
local success = Xml.IntoElement( parser )
```

See Also

- Xml.OutOfElement

Xml.OutOfElement

Brief

Navigate out of the current XML element.

Definition

```
boolean Xml.OutOfElement( Xml parser )
```

Arguments

parser - The instance of the XML parser.

Return Values

true it was possible to navigate out of the current element.

Description

Navigate out of the current XML element.

Examples

```
local success = Xml.OutOfElement( parser )
```

See Also

- Xml.IntoElement

Xml.SetData

Brief

Sets the data for the XML parser.

Definition

```
Xml.SetData( Xml parser, string resourceName )
Xml.SetData( Xml parser, Resource resource)
```

Arguments

parser - The instance of the XML parser.resourceName - The name of the resource in the object containing the XML data.resource - The resource containing the XML data.

Return Values

None.

Description

Sets the data for the XML parser. The specified resource may be the resource of an object or it may be loaded via the Resource library from a URL.

Examples

```
-- Example 1: Set the parser to an XML resource in the object
parser = Xml.Create()
Xml.SetData( parser, "settings" )

-- Example 2: Set the parser to a downloaded XML resource
res = Resource.Request( "http://www.example.com/data.xml", "xml")

-- Do some work while XML resource is downloaded

if ( Resource.IsLoaded( res ) ) then
    Xml.SetData( parser, res )
end
```

See Also

- Resource.IsLoaded
- Resource.Request
- Xml.Create

Xml.SetDataFromContainer

Brief

Sets the data for the XML parser from a memory container.

Definition

```
Xml.SetDataFromContainer( Xml parser, MemoryContainer container )
```

Arguments

parser - The instance of the XML parser.container - The container that holds the XML data.

Return Values

None.

Description

Sets the data for the XML parser from a memory container. This is for testing XML output from the BasicGenx library and has no real practical use.

Examples

```
-- Write some XML to a memory container
container = MemoryContainer.Create( 1024 )
writer = BasicGenx.Create( container )
BasicGenx.StartDocument( writer )
BasicGenx.StartElement( writer, "XML" )
BasicGenx.EndElement( writer )
BasicGenx.EndDocument( writer )

-- Set the XML reader to the output from BasicGenx
Xml.SetDataFromContainer( parser, container )
```

See Also

- BasicGenx.Create
- MemoryContainer.Create